BINARY DATA RECONSTRUCTION IN PRIVACY-PRESERVING RECOMMENDATION ALGORITHMS

Ph.D. Dissertation

Murat OKKALIOĞLU

Eskişehir 2017

BINARY DATA RECONSTRUCTION IN PRIVACY-PRESERVING RECOMMENDATION ALGORITHMS

Murat OKKALIOĞLU

Ph.D. Dissertation

Computer Engineering Program
Supervisor: Assoc. Prof. Dr. Cihan KALELİ

Eskişehir
Anadolu University
Graduate School of Sciences
November 2017

This dissertation is supported by TUBİTAK under grant 113E262

FINAL APPROVAL FOR THESIS

This thesis titled "Binary Data Reconstruction In Privacy-Preserving Recommendation Algorithms" has been prepared and submitted by Murat OKKALIOĞLU in partial fullfillment of the requirements in "Anadolu University Directive on Graduate Education and Examination" for the Degree of Doctor of Philosophy (PhD) in Computer Engineering Department has been examined and approved on 13/11/2017.

Committee Members		**Signature**
Member (Supervisor)	: Assoc. Prof. Dr. Cihan KALELİ	……
Member	: Asst. Prof. Dr. Alper BİLGE	……
Member	: Asst. Prof. Dr. Mehmet KOÇ	……
Member	: Asst. Prof. Dr. Alper Kürşat UYSAL	……
Member	: Asst. Prof. Dr. Uğur GÜREL	……

……

Director

Graduate School of Sciences

ABSTRACT

BINARY DATA RECONSTRUCTION IN PRIVACY-PRESERVING RECOMMENDATION ALGORITHMS

Murat OKKALIOĞLU

Department of Computer Engineering

Anadolu University, Graduate School of Sciences, November 2017

Supervisor: Assoc. Prof. Dr. Cihan KALELİ

Collaborative filtering systems have become very popular with the frequent use of the Internet to offer reliable recommendations to users. Ratings for such systems could be in a binary or numeric scale, and data supplied by users could be stored in a central-server, distributed among two- or multi-party or even peers could come together for collaborative filtering purposes. Collaborative filtering systems rely on true user feedbacks in order to produce accurate recommendations. However, users of such systems might be reluctant to provide their true opinions if they feel that their confidential data might be used other than the initial purpose of data collection. Such resistances to participate in collaborative filtering systems might hamper the recommendation quality. At this point, privacy-preserving collaborating filtering systems take the privacy concerns into the primary consideration without sacrificing the recommendation quality. Therefore, users are convinced to provide their true inputs as well as receive quality recommendations by the measures taken by privacy-preserving collaborative filtering systems. However, these measures should be investigated if the claimed privacy-preservation is really maintained. The objective of this dissertation is to derive the original binary ratings, which are promised to be preserved, from the perturbed binary ratings in different privacy-preservation protocols under different data partitioning scenarios including central server-based, distributed between two- and multi-party and peer-to-peer collaboration. Auxiliary information is utilized throughout the dissertation to improve the reconstruction accuracy or circumvent the bottlenecks to derive the original ratings due to the privacy-preservation protocols.

Keywords: Privacy, binary ratings, data reconstruction, auxiliary information, collaborative filtering.

ÖZET

GİZLİLİK TABANLI ÖNERİ ALGORİTMALARINDA İKİLİ VERİLERİN YENİDEN OLUŞTURULMASI

Murat OKKALIOĞLU

Bilgisayar Mühendisliği Anabilim Dalı

Anadolu Üniversitesi, Fen Bilimleri Enstitüsü, Kasım 2017

Danışman: Doç. Dr. Cihan KALELİ

Ortak filtreleme sistemleri Internet' in sıklıkla kullanılmasıyla beraber kullanıcılara güvenilir tavsiyeler üretmek için çok popüler oldu. Bu sistemlerin oylamaları ikili veya nümerik bir ölçekte olabilir ve kullanıcılar tarafından sağlanan veriler merkezi bir sunucuda tutulabilir, iki- veya çok-parti arasında dağıtık olabilir ve hatta eşler ortak filtreleme amaçları ile bir araya gelebilirler. Ortak filtreleme sistemleri doğru tavsiyeler üretebilmek için kullanıcıların doğru geri bildirimine bel bağlarlar. Fakat, bu sistemlerin kullanıcıları kişiye özel verilerinin toplanma amacı dışında kullanılabileceğini hissederlerse gerçek fikirlerini sağlamakta isteksiz davranabilirler. Ortak filtreleme sistemlerine katılmak için böyle bir direniş tavsiye kalitesini aksatabilir. Bu noktada, gizlilik tabanlı ortak filteleme sistemleri gizlilik endişelerini tavsiye kalitesini feda etmeden öncül olarak göz önüne alırlar. Bu yüzden, kullanıcılar gizlilik tabanlı ortak filtrelemede alınan önlemler sayesinde kaliteli tavsiye almanın yanında doğru girdiler sağlamaya da ikna edilirler. Fakat, bu önlemlerin ifade edilen gizlilik korumasını sağlayıp sağlamadığı incelenmelidir. Bu tez çalışmasının amacı merkezi sunucu tabanlı, iki- veya çok-parti arasında dağıtılmış ve eşler arası işbirliğini içeren farklı veri dağıtımı senaryoları altında farklı gizlilik koruma protokolleriyle saklanmış ikili oylamalardan korunması sözü verilen orijinal ikili oylamaların elde edilmesidir. Veri imarının doğruluğunu ve gizlilik koruma protokollerinden orijinal ikili oylamaları elde ederken karşılaşılan engellerden kurtulmak için yardımcı bilgi kullanılmıştır.

Anahtar kelimeler: Gizlilik, ikili oylamalar, veri imari, yardımcı bilgi, ortak filtreleme.

ACKNOWLEDGEMENTS

I would like to express my sincerest appreciation to my supervisor, Assoc. Prof. Dr. Cihan Kaleli, for his help, guidance and support throughout the path toward this dissertation. It was an honor and a great experience to be able work with him.

I am grateful to Asst. Prof. Dr. Mehmet Koç and Asst. Prof. Dr. Alper Bilge for serving in my dissertation monitoring committee and for their support and encouragements throughout the dissertation. Also, I am grateful to Asst. Prof. Dr. Alper Kürşat Uysal and Asst. Prof. Dr. Uğur Gürel for serving in my dissertation committee.

I believe words cannot express my appreciation and gratitude to my wife, Burcu, for her patience and support. Last but not least, I am grateful to my parents and sisters for their unconditional love and values that shape me to be who I am today as a person.

Murat OKKALIOĞLU

November, 2017

13/11/2017

STATEMENT OF COMPLIANCE WITH ETHICAL PRINCIPLES AND RULES

I hereby truthfully declare that this thesis is an original work prepared by me; that I have behaved in accordance with the scientific ethical principles and rules throughout the stages of preparation, data collection, analysis and presentation of my work; that I have cited the sources of all the data and information that could be obtained within the scope of this study, and included these sources in the references section; and that this study has been scanned for plagiarism with "scientific plagiarism detection program" used by Anadolu University, and that "it does not have any plagiarism" whatsoever. I also declare that, if a case contrary to my declaration is detected in my work at any time, I hereby express my consent to all the ethical and legal consequences that are involved.

Murat OKKALIOĞLU

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

δ_{AP} : Filling Factor for Active Peer

δ_{AU} : Filling Factor for Active User

θ : Randomized Response Threshold Value

π : Percentage of Sensitive Attribute

τ : Predefined Threshold

AP : Active Peer

AU : Active User

AK-ICA : A-priori Independent Component Analysis

CA : Classical Approach

CF : Collaborative Filtering

d : Density

d_i : Number of Dislike

d_{set} : Density of the data set

DV : Default Voting

EM : Expectation Maximization

FA : Fair Approach

G : Number of Groups

HDD : Horizontally Distributed Data

HPD : Horizontally Partitioned Data

HRI : Hiding Rated Items

ICA : Independent Component Analysis

IMDB : The Internet Movie Database

knn : k-Nearest Neighbor

l_i : Number of Likes

m : Number of Items

MLM : Movie Lens Million

n : Number of Users

N_{AU} : Active User's List of Items for Recommendation

NBC : Naïve Bayes Classifier

P2P : Peer to Peer

P3CF : Privacy-Preserving Partitioned Collaborative Filtering

PCA : Principal Component Analysis

PD : Party's Denominator

PN : Party's Nominator

PPDCF : Privacy-Preserving Distributed Collaborative Filtering

PPCF : Privacy-Preserving Collaborative Filtering

PPDM : Privacy-Preserving Data Mining

PPP : Peer/Party Privacy for Parties

PPR : Peer/Party Privacy for Ratings

PSCP : Private Similarity Computation Protocol

prec : Precision

q : Queried Item

rec : Recall

RF : Random Filling

RRT : Randomized Response Technique

SF : Spectral Filtering

SVD : Singular Value Decomposition

SW : Significance Weight

top-N : Top-N Recommendations

VDD : Vertically Distributed Data

VPD : Vertically Partitioned Data

To my parents

1. INTRODUCTION

The Internet has created new forms of interactions. Earlier daily routines such as shopping, ordering a meal, financial transactions or face-to-face conversations have been already transferred to online media. International Telecommunication Union (2017) estimates that 48% of world population and 70.6% of youth use the Internet in 2017. On the one hand, the amount of data created online has been steadily growing (McAfee et al., 2012) due to such a frequent and widespread use of the Internet. On the other hand, people utilizing online solutions might be overwhelmed while deciding due to the abundance of options with which they are faced. This phenomenon, which requires processing more information than one can handle to make a decision, is called *information overload*. E-commerce companies would like to attract more customers by finding out which products would best fit their customers' tastes to overcome information overload problem. At this point, *collaborative filtering* (CF) is a technique to offer personalized recommendations based on the item or user preferences.

1.1. Collaborative Filtering

CF is a technique that aims to provide recommendations to its users by utilizing earlier preferences of items. The basic idea behind CF is that like-minded users have similar tastes so that they should be offered similar items. CF only uses feedbacks provided by users. A CF system offers predictions for a user by matching him or her with other users with similar tastes. This technique helps users reduce the time to search for a right product. The term was first coined by Goldberg et al. (1992) for an e-mail filtering system, Tapestry, which users annotate their emails for recommendations. A traditional CF system operates on a large $n \times m$ rating matrix, where n different users, $\{u_1, u_2, \ldots, u_{n-1}, u_n\}$, rate any of m different items, $\{it_1, it_2, \ldots, it_{m-1}, it_m\}$. A rating matrix is usually sparse because users can only rate some of the items among a very large data set. In general, CF systems rely on either *user-based* or *item-based* algorithms. In the user-based CF systems, the *active user* (*AU*) who is looking for a prediction is matched with some other users called *neighbors* based on a metric measuring the similarity. A prediction is determined based on ratings from neighbors presuming that items liked by neighbors will also be liked by an AU (Herlocker and Konstan, 1999). On the other hand, item-based algorithms explore relationships between items instead of users to avoid searching users among (Sarwar et al., 2001). CF systems utilize numeric (scalar) or binary ratings.

Numeric ratings indicate how much an item is preferred by a user from a discrete or continuous value scale between two numbers. Binary ratings express if a user likes/agrees or dislikes/disagrees an item. Ratings can also be categorized as explicit and implicit. While explicit ratings are collected from users and represent users' preferences, implicit ratings are more related to an inference from users' behaviors such as browsing, purchase, or transaction history (Schafer et al., 2007). Implicit ratings can be referred as unary ratings indicating presence or absence. There are different CF schemes utilizing implicit (Oard and Kim, 1998; Hu, Koren and Volinsky, 2008) or explicit ratings (Breese, Heckerman and Kadie, 1998; Herlocker and Konstan, 1999; Miyahara and Pazzani, 2000). An example of a rating matrix is given in Figure 1.1 to illustrate a CF system with binary ratings. In Figure 1.1, user *Cihan* wants a prediction for the book *Pinocchio*, which is question-marked in the figure. A prediction will be made among the like-minded or like-rated items based on the CF prediction algorithm. Light grey users denote the neighbors of *Cihan*, and the prediction will be made between these two users if a user-based algorithm is utilized. If an item-based algorithm is used, light gray items will be picked as neighbor items, and the prediction will be produced based on these items.

	The Lord of the Rings	The Little Prince	The Hobbit	A Tale of Two Cities	The Exorcist	The Plague	The Hunger Games	Pinocchio
Burcu	0	-	0	-	1	1	-	1
Gul	0	1	-	-	-	1	0	1
Salih	1	1	-	0	0	0	-	-
Esra	-	-	1		0	-	1	0
Cihan	1	-	1	0	0	-	1	?

Figure 1.1. *An example of a CF matrix*

CF methods can be categorized into three different groups, namely, *memory-based, model-based* and *hybrid* CF. In memory-based methods, all rating matrix is utilized to produce recommendations. These methods usually need some sort of a similarity weight to obtain the neighborhood. Upon generating the neighborhood, recommendation algorithm utilizing the neighborhood information is executed. In memory-based CF methods, correlation-based and vector-based similarity calculations are widely preferred (Resnick et al., 1994; Konstan et al., 1997; Breese et al., 1998; Herlocker and Konstan, 1999). Model-based CF utilizes a model, which could be a machine learning algorithm or a data mining method, on training data to learn. Then, predictions are generated based on

the model learned (Breese et al., 1998). Hybrid methods are the combination of memory- and model-based methods, but it might include the use of textual information as well (Su and Khoshgoftaar, 2009).

Until now, *recommendation* and *prediction* have been used to express CF systems' output generated for AU based-on AU's past preferences (rating vector). In CF context, *prediction* refers to predict a rating for a given item asked by AU. *Recommendation* refers to a list of items in which AU could be interested. Also, there might be cases that AU could specify set of items that she wants to be recommended. For example, in a music-related CF system, assume that AU is a classical-music listener and she might not prefer to be recommended from other genres. Such a case is called *constrained recommendations*, AU could specify items from which her recommendations should be generated (Schafer et al., 2007).

There are some challenges that CF systems should take care of. E-commerce companies might have a large variety of items so that users usually do not have an idea for most of the items in the data set, which causes sparsity. Data sparsity is one of the main challenges affecting the quality of recommendations (Sarwar et al., 1998). Since CF algorithms try to find out correlations between users or items, stronger relationships can be extracted with denser datasets. Data sparsity could lead to some problems other than recommendation quality. When a new user or item is inserted into data set, it would be difficult to find similar users or items because they have either limited or no ratings in their vectors. This problem is known as *cold start*. *Reduced coverage* and *neighbor transitivity* are the other problems encountered due to data sparsity (Su and Khoshgoftaar, 2009). Reduced coverage occurs when CF algorithm cannot produce recommendations for some users due to relatively small rated items. Moreover, if a CF system cannot identify any neighbor for AU, this problem is called neighbors transitivity because no user have rated any common item with AU. Due to problems stemming from data sparsity, a denser data set is desirable for a more reliable CF system.

Scalability is a general term to define the capability of a system to keep up with increasing volume of work. Since CF systems are usually available online, they are bound to frequent user interactions and data matrix is obliged to grow. Therefore, CF algorithms should cope with the growing size of data while offering predictions. Dimension reduction techniques could be employed to reduce the size of a large data matrix. Apart

from scalability and sparsity, *grey sheep* and *synonym* are the challenges for CF (Su and Khoshgoftaar, 2009).

Up to now, operational challenges about CF is given. Above all else, users must provide their true opinion so that quality predictions can be produced. The key factor for a CF system to function properly regarding prediction accuracy is the voluntary and true user participation. To achieve this goal, users must be assured that their preferences would not be comprised. Therefore, privacy is another challenge for CF systems that could dramatically affect the true user participation. Users might be unwilling to share their opinions because they could go widely public if disclosed (Resnick and Varian, 1997). Ratings made by users can disclose opinions on sensitive issues or can be compromised against them. Therefore, users of a CF system might be reluctant to participate or mislead a CF system by providing false opinions if they believe that their privacy could be compromised. Data is a valuable asset and it can be sold in case of bankruptcy (Canny, 2002). Furthermore, unsolicited marketing, government surveillance, price discrimination or subpoena are various examples of invasion of privacy that could be exploited by secondary use, which is the use of other than the initial purpose of data collection, of individual data (Cranor, 2003; Culnan, 1993). *Privacy-preserving collaborative filtering* (PPCF) is a technique that puts special emphasis on privacy without neglecting the prediction accuracy. Therefore, PPCF bestows a right for privacy for users and aims to provide an equilibrium between privacy and accuracy because they are conflicting goals (Polat and Du, 2005b).

1.2. Privacy and Privacy-Preserving Collaborative Filtering

1.2.1. Privacy and Internet

In 1879, Judge Cooley discussed a term, *personal immunity*, as "the right to one's person may be said to be a right of complete immunity: to be let alone (Cooley, 1879)" as one of the legal rights of a person. In 1890, Warren and Brandies (1890) published an article concerning whether law offers a privacy protection for individuals if so, the extent and nature of it. In their article, protection against threats to a person and property is very old in law. However, recognition of different rights is indispensable owing to changes in life. Warren and Brandies (1890) argue that property is nothing more than any form of possession either tangible or intangible and recent developments highlight the need for the right "to be left alone" by citing Judge Cooley's personal immunity definition (Cooley

1879) as a right to privacy. Solove (2002) gives six headings to conceptualize privacy. These include a variety of circumstances from "the right to be let alone" (Warren and Brandies, 1890), to limited access to self, control over personal information, secrecy, personhood, and intimacy. Magi (2011) discusses why privacy is important in fourteen reasons and lays out three main reasons: the benefits of privacy for individuals, inter-personal relations and society. In these categorizations (Magi, 2011; Solove, 2002), one must decide own space for privacy. A disclosure of any matter considered private could cause many undesired circumstances. For example, one could be subject to a personal profiling, being misjudged or power imbalance between individuals and intuitions (Magi, 2011). Therefore, privacy is a factor that can affect many aspects of one's personal life and one could manage what is communicated about him- or herself to others (Westin, 1967 as cited in Solove, 2002). As with this aspect of privacy, *information privacy* is defined as

> "refers to the claims of individuals that data about themselves should generally not be available to other individuals and organisations, and that, where data is possessed by another party, the individual must be able to exercise a substantial degree of control over that data and its use Clarke (1999, p. 60)".

Privacy concerns of e-commerce users have been grouped into three categories as privacy fundamentalists, pragmatic majority and marginally concerned by Ackerman, Cranor and Reagle (1999). *Privacy fundamentalists* are the ones who are very concerned about their privacy with 17% while *marginally concerned* are ready to give any input with a mild consideration of privacy. This group constitutes 27% of the users. On the other hand, *pragmatic majority* is the group of people with 56% who have less privacy concern than fundamentalists but marginally concerned ones. Their concern could be alleviated by acknowledging privacy measures (Ackermann, Cranor and Reagle, 1999). Users might also waive their privacy priorities if they are offered benefits in return (Berendt, Günther and Spiekermann, 2005). This study investigates if stated privacy of users is in accordance with their online behavior. In the study, incentives are offered to the users with the assistance of a shopping bot in order to test if they are willing to disclose their information. Participants are grouped into four groups. In addition to privacy fundamentalists (30%) and marginally concerned (24%) recognized by Ackerman, Cranor and Reagle (1999), Berendt, Günther and Spiekermann (2005) come up with identity concerned (20%) and profiling averse (26%) groups. *Identity concerned* people

are more cautious about their personal information such as name, birth date, and age to be disclosed while *profiling averse* ones are more concerned about being categorized based on information such as hobbies, health status, and political view. The study shows that many users do not take their stated privacy into consideration due to potential benefits in return. Therefore, Berendt, Günther and Spiekermann (2005) discuss that privacy statements have no impact on behavior. In terms of privacy, users are inclined to act differently from what they initially stated. Paine et al. (2007) perform a survey with open-ended questions to understand privacy perception of users. Per their study, 56% of respondents declare that they have privacy concerns when they are online and 73% of respondents take actions to protect their privacy. Also, most respondents with privacy concerns take actions to protect their privacy. The ones who are concerned about their privacy yet take no action state that they just do not know what to do (Paine et al., 2007). Bélanger and Crossler (2011) present a comprehensive study that covers information privacy literature in detail.

In terms of CF, some people might be unwilling to share their true opinions due to a variety of privacy concerns. Friedman et al. (2015) state that privacy risks might occur either with direct access to data or inference of new data and they define three types of adversaries, the recommender system itself, other users or external entities. The recommender system might misuse personal information; other users might exploit CF outputs, and external entities such as hackers or legal entities might access data. On the other hand, PPCF primarily focuses on protecting privacy which would help encourage users to participate in a recommendation process by assuring them with a privacy protocol instead of privacy statements to protect their data. The main idea behind a typical PPCF scheme is that users perturb their data before sending it to the server. Privacy is usually performed by perturbing original data; therefore, it comes at the cost of losing some information due to perturbation. As privacy measures are tightened to increase the level of privacy, the outcoming data might become very different from the original one to provide accurate predictions. Thus, privacy and accuracy are both need to be considered without sacrificing one to another. Thus, it is a fundamental tradeoff that must be addressed in PPCF.

1.2.2. Data partitioning and PPCF

PPCF schemes might differ according to the type of data storage. Data could be stored centrally, shared between two- or multi-parties. Behind these, decentralized peer-to-peer (P2P) schemes are also available. In *central server-based* PPCF schemes, data is held by a central server that is going to provide predictions based on ratings provided by users. In such a scheme, users usually send their ratings to the central data holder after applying a perturbation method so that the data holder does not have the original ratings. Some companies that want to grow their business or get into a new market might not have enough ratings to produce accurate predictions. Such companies could come together to enhance their rating matrices (Polat and Du, 2005c). They can share their matrices either vertically or horizontally. Sharing data horizontally means that corresponding parties share ratings of the same set of items from different users. Sharing data vertically means that corresponding parties share ratings of the same set users for different items. By doing so, a party in this process obtains new ratings for its users. Imagine two different movie rental companies utilizing a CF algorithm to promote the sales. One of them would like to grow in its own sector while the other would like to branch out into a bookstore. The first company would choose to cooperate with another movie rental company that has different customers. Such cooperation, which adds new users to the same set of movies for both parties, results in a denser matrix for the first movie rental company, which wants to grow in the sector. This kind of data sharing is called *horizontally-partitioned data* (HPD). If the second movie rental company shares its movie ratings with a bookstore and obtains other party's book ratings for its users in return, this cooperation adds new rating variety for the users of both parties. Such cooperation is called *vertically-partitioned data* (VPD), and it helps enterprises that are interested in new markets. If HPD or VPD is performed between multi parties, it is called *horizontally-* (HDD) or *vertically-distributed data* (VDD). Figure 1.2 displays HDD- and VDD-based data sharing between parties. If data sharing is between two-parties with privacy, it will be hereafter called *privacy-preserving partitioned collaborative filtering* (P3CF) (Bilge et al., 2013). Likewise, if it is between multi-parties with privacy considerations, then it will be called *privacy-preserving distributed collaborative filtering* (PPDCF). Besides any server-based schemes either central, two- or multi-party, users (peers) might act individually to collaborate in a PPCF process. Peers can participate in a P2P network for CF purposes

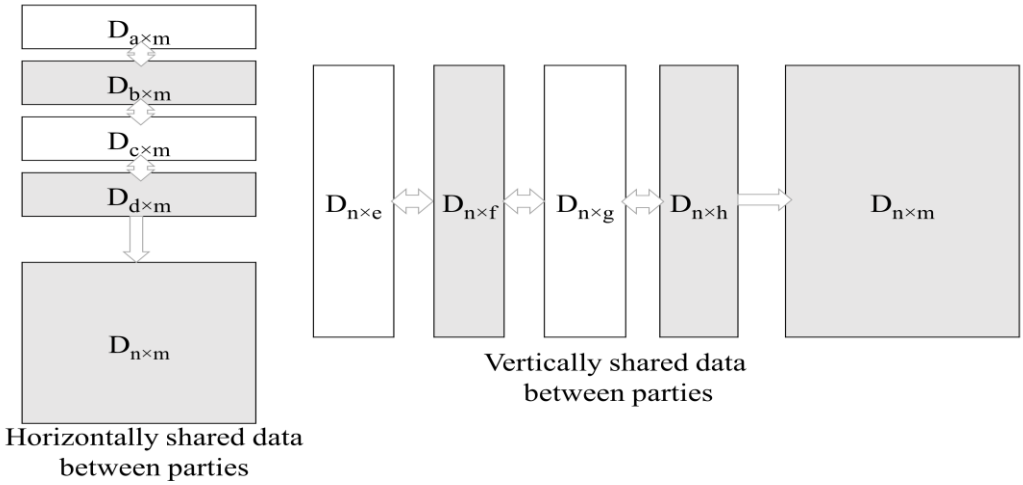


Figure 1.2. *HDD- and VDD-based data sharing*

with privacy. Such a scheme does not require any server; predictions are produced by a collaborative effort of peers. Such schemes without any data holder are called P2P PPCF.

In PPCF, there are some techniques to perturb user data. *Randomized perturbation* is a widely-used one for the purpose of PPCF. This method adds a random noise to an original numeric rating so that the original data can be kept private to the extent of the appended noise. Traditionally, the random number, *r*, is added to the original rating, *x*. Polat and Du (2003; 2005a) are the first employing randomized perturbation to disguise original ratings of the users. The users calculate *z*-scores for each item and disguise them by adding random numbers drawn from either uniform or normal distribution before sending their vectors to the server. Each user might have different concern for privacy. Therefore, these concerns might be addressed by inconsistently disguising each user vector according to their needs (Polat and Du, 2007). Polatidis et al. (2017) propose to add a level to the randomization allowing each user to generate a random perturbation level. Matrix factorization techniques such as singular value decomposition (SVD) are employed with randomized perturbation to offer privacy for users (Polat and Du, 2005d; Yakut and Polat, 2010).

Obfuscation is a method to substitute real ratings with fake ones. Berkovsky et al. (2005) propose this method claiming that parts of users' data can be obfuscated. The authors propose three methods of substitutes which are default, uniform and bell curved. As the names imply, users substitute their ratings with a default predefined value while random numbers drawn from uniform and normal distributions are substituted in the uniform and bell curved methods, respectively. This study is taken one step further by

introducing hierarchical neighborhood (Berkovsky et al., 2007). The authors organized users (peers) in groups and groups are managed by a superuser who organizes communication among other groups. Privacy in groups is maintained by obfuscation (Berkovsky et al., 2005). Berkovsky, Kuflik and Ricci (2012) show the effect of applying obfuscation on extreme and overall items claiming that prediction of extreme items is more important than overall items for users.

The studies mentioned above are designed for numeric ratings. Users can also express their tastes in a binary scale. *Randomized response technique* (RRT), which is a survey technique proposed by Warner (1965) to find out the prevalence of a sensitive attribute in a population, is employed on binary ratings to disguise ratings of users in a central-server based PPCF scheme (Polat and Du, 2006). The idea of applying RRT is to reverse or preserve whole or part of a user vector based on a random determiner. Each user picks a random number and compares it with a predefined threshold to keep or reverse a rating vector. The details of this method and RRT are given in Chapter 2 where preliminary information is introduced. RRT is applied on P3CF with HPD- and VPD-based schemes (Polat and Du, 2005c; 2008) and these schemes can be easily adapted into HDD- and VDD-based PPCF schemes. As a prediction algorithm, Naïve Bayes classifier (NBC) is frequently used when ratings are binary. Miyahara and Pazzani (2000) use NBC for CF purposes. Kaleli and Polat (2007b) use RRT to provide privacy and utilize NBC for predictions. Their method covers a central server-based PPCF scheme. These NBC-based prediction schemes are extended into P3CF (Kaleli and Polat, 2007a), PPDCF (Kaleli and Polat, 2015) and P2P PPCF(Kaleli and Polat, 2010). The above-mentioned binary PPCF schemes are handled in detail in Chapter 2 as well. Although PPCF has been receiving increasing attention over the last decade, the field is open for improvement, and overall performance should be improved (Ozturk and Polat, 2015). In this respect, Kaleli and Polat (2009) and Bilge and Polat (2010) put efforts to improve the performance of NBC-based predictions. Kaleli and Polat (2009) cluster users while Bilge and Polat (2010) form a neighborhood by determining the best similar item for each item with Tanimoto coefficient as binary similarity measure. The authors also fill vectors to increase the density of the dataset.

In PPCF, privacy is considered in two aspects (Polat and Du, 2006). Given that a user has a rating for some items in a traditional CF user-item matrix, the first aspect of privacy is about preserving the exact rating value made for an item. The second aspect of

privacy is related to whether an item is rated or not. The first aspect of privacy is trivial because the main idea is that users do not want their ratings to be disclosed explicitly. The second aspect of privacy deals with profiling of a user because users could be socially, politically or sexually profiled based on the knowledge of which items they rate although their explicit ratings are not known. Privacy has another dimension apart from these two aspects. In central server-based and P2P PPCF schemes, users or peers want to preserve their confidential data from the server or other peers, respectively. Therefore, they would prefer to employ perturbation methods to avoid data disclosure. If users or peers take privacy measures to prevent from any disclosure of confidential information, it is called *individual privacy* (Bilge et al., 2013). On the other hand, an e-commerce company that would like to collaborate with other companies in a two- or multi-party fashion should preserve the privacy of its users. Such a company could employ data perturbation methods on its own data to avoid the disclosure of confidential information of its users. In two- or multi-party PPCF schemes, users usually send unperturbed data to their server. Therefore, companies in two- or multi-party collaboration should prevent their data from other parties. This type of privacy preservation is called *institutional* or *corporate privacy* (Bilge et al., 2013).

1.3. Problem Definition

Although individual or institutional privacy is promised to be maintained by PPCF schemes, these promises need to be investigated whether PPCF schemes are indeed immune to different attempts to derive original private data. In *privacy-preserving data mining* (PPDM) literature, the scholars (Huang, Du and Chen, 2005; Liu, 2007; Liu, Giannella and Kargupta, 2006; Guo and Wu, 2007) demonstrate that privacy might not be preserved. Inspired from such studies, PPCF schemes should be also scrutinized whether or how much privacy is provided by different PPCF schemes. Ratings could be either numeric or binary form. Studies in PPCF community to derive confidential data is mainly focused on numeric ratings (Zhang, Ford and Makedon, 2006; Demirelli Okkalioglu, Koc and Polat, 2016). Deriving original binary ratings from PPCF schemes under different data partitioning scenario is not as much as studied. The concentration in this dissertation is to derive original private binary ratings from PPCF schemes where data is stored in a central server, shared between two- or multi-party or distributed between peers. A malicious adversary might attempt to target individual or institutional

privacy while operating a PPCF protocol. This task is usually accomplished by proposing different reconstruction attacks that exploit the weakness in PPCF protocols.

The main objective while deriving original data from a central server-based scheme is to disclose users' data. Users send their data after they perturb it; therefore, the server gets the perturbed version of user vectors. In the presence of a semi-trusted server, users' data might be at risk. A semi-trusted server is the one who acts in accordance with the PPCF protocol; however, it might exploit flaws of the protocol that may cause information disclosure to derive confidential information. The malicious server targets individual privacy of its users. In this dissertation, a malicious server will contemplate an attack technique to derive original user vectors in a central server-based PPCF scheme.

When data is partitioned or distributed between parties, different data holders might come together to diversify their rating matrices with new users or items. In partitioned or distributed cases of PPCF, parties hold original user vectors as institutional data. Parties need to keep their institutional data private while collaborating with other parties. The main scenario when data is distributed among different parties is to derive institutional data of other parties when there occurs a semi-honest party who would exploit weaknesses in the protocol while fulling its responsibilities.

The last case deals with P2P PPCF schemes where peers collaborate for private prediction purposes. However, one of the peers might act maliciously to derive the confidential data of others. The malicious peer who exploits the PPCF scheme while performing the protocol requirements could target the individual privacy of other peers. Although P2P collaboration is a distributed scheme, the main difference between P2P PPCF and PPDCF is that a malicious peer targets individual privacy of other peers while a malicious party targets institutional privacy of other parties.

1.4. Related Work

Deriving private information from perturbed data has been studied in PPDM to examine different privacy-preservation techniques. Scholars developed various methods to recover original data to show how well the original data has been hidden. On the other hand, the efforts made to derive private information in PPCF schemes are still limited when compared to PPDM literature. In this part, such attacks in PPDM and PPCF are grouped into five major classes, which are spectral filtering (SF)-based, principal component analysis (PCA)-based, singular value decomposition (SVD)-based,

independent component analysis (ICA)-based, and other attacks. The first three classes of attacks are highly related to each other.

1.4.1. Spectral Filtering-based reconstruction

One of the primary methods for deriving original data from random perturbation or randomization is to utilize SF. Randomization is a frequently used technique to protect numerically rated user data (Agrawal and Srikant, 2000). In this setting, a random value, r, is picked from a distribution (Gaussian or uniform) and it is added to the original value, x_i, to get a substitute value, $x_i + r$, which will take the place of x_i in the data set. The studies presented in (Kargupta et al., 2003; 2005) discuss the theoretical bounds of maximum and minimum eigenvalues of a random matrix for random perturbation. After determining maximum and minimum eigenvalues of a random matrix, the noise created by the random matrix can be filtered off. During their experiments, the authors also introduce signal-to-noise ratio that quantifies how much noise added to the signal (original data). Dutta et al. (2003) extend the work in (Kargupta et al., 2003) by considering various data types. Experiments show that the estimation accuracy decreases as the amount of noise increases, which is associated with low signal-to-noise ratio. Guo and Wu (2006) study on determining the success of the SF-based reconstruction. They derive an upper bound for the attacker to assess how close the estimate to the original data when spectral filtering is exploited. A similar observation stated by Guo, Wu and Li (2008) in terms of upper bound for reconstruction error; however, they also determine a lower bound for data owners to specify how much noise should be added for desired privacy. An SVD-based approach is used for lower bound extraction; however, it is indeed equivalent to SF lower bound.

1.4.2. PCA-based reconstruction

PCA is a technique to express given data in reduced number of dimensions by exploiting correlation among data. PCA-based reconstruction approach, introduced by Huang, Du and Chen (2005), reconstructs the original data from the disguised data perturbed by randomization. The authors show that PCA-based method reconstructs accurately in highly correlated data. On the other hand, they introduce a different version of randomization that adds correlated noise to enhance privacy. PCA is also recognized by other researchers to obtain original data (Liu, Giannella and Kargupta, 2006; Turgay et al., 2008) and these studies assume that the attacker has some prior knowledge about

data. The scholars (Liu, Giannella and Kargupta, 2006) assume that the attacker has known inputs-outputs and some known samples. Likewise, Turgay et al. (2008) assume that the attacker has a dissimilarity matrix. The PCA-based attack is built on top the hyper-lateration technique, where a candidate data set is generated from the dissimilarity matrix.

1.4.3. SVD-based reconstruction

SVD-based reconstruction attacks are highly related to SF-based ones. SVD splits a matrix into three matrices as follows: $A_{n\times m} = U_{n\times n}S_{n\times m}V^T_{m\times m}$, where $U_{n\times n}$ and $V^T_{m\times m}$ are orthogonal matrices and $S_{n\times n}$ is a diagonal matrix with singular values of A. An upper bound is determined with SF-based reconstruction (Guo and Wu, 2006). The scholars (Guo and Wu, 2006; Guo, Wu and Li, 2006; 2008) give an upper and lower bound on reconstruction error for SF-based reconstructions. Upper bound can be used by attackers to determine how close their estimation to the original data. The lower bound can be used by data holders to arrange privacy level of their data. They also prove the equivalence of SF and SVD. In this method, the important point is to find the first k singular values of perturbed data. Another SVD-based reconstruction with expectation maximization (EM) is proposed by Zhang, Ford and Makedon (2006) to derive numeric data from masked data.

1.4.4. ICA-based reconstruction

The reconstruction methods covered until now are designed against randomized perturbation. Nonetheless, there are also other data reconstruction methods worth mentioning. Some studies focus on recovering data perturbed by multiplicative perturbation like *rotation perturbation* (Chen and Liu, 2005; Oliveira and Zaïane, 2010) and *random projection* (Liu, Kargupta and Ryan, 2006). Such perturbations are defined as $Y = MX$, where M is a mixing matrix. ICA is a technique to observe a linear representation of statistically independent components. In rotation perturbation case, when Y is observed, both M and X can be estimated with some restrictions via ICA method. Hyvärinen, Karhunen and Oja (2001) mention three restrictions and two ambiguities about ICA. Two of these restrictions are argued by researchers (Chen and Liu, 2005; Liu, Kargupta and Ryan, 2006). The authors claim that ICA is ineffective due to the restrictions (source signals are independent, and all of them must be non-Gaussian except one). These restrictions are not very applicable in privacy-preserving data mining

approaches. In addition to these restrictions defined in ICA, two ambiguities, the order of recovered data is not guaranteed (Chen and Liu, 2005; Chen, Sun and Liu, 2007) and variances of original data cannot be determined, make ICA ineffective if additional statistics about data is not known (Chen, Sun and Liu, 2007). A-priori knowledge ICA (AK-ICA) attack is utilized when a sample of original data is available (Guo and Wu, 2007). AK-ICA applies ICA on both perturbed data and sample of original data; then it explores relationships among them to reconstruct original data. Undetermined ICA is applied to perturbation methods when the mixing matrix is rectangular, and the mixing matrix is known with prior knowledge (Sang, Shen and Tian, 2009; 2012).

1.4.5. Other attacks

Agrawal and Srikant (2000) are the pioneers that aim to provide privacy in data mining. They state that original data distribution can be reconstructed after data is perturbed by randomization. Their aim is to estimate the original distribution not to reconstruct individual values. The authors apply Bayes' rule to show that the distribution of the original data can be estimated. Agrawal and Aggarwal (2001) utilize EM to estimate original distribution. If a large amount of data is available, EM can produce a good estimate of the original distribution. The scholars also propose a quantification of privacy and information loss stating that increased privacy causes information loss. Huang, Du and Chen (2005) utilize Bayes estimated-based data reconstruction. As the name implies, they search for reconstructed data X given perturbed data Y that maximizes $P(X|Y)$. While above approach targets randomization, Liu (2007) designs a similar attack, maximum a posteriori probability attack, against random projection. Zhang, Ford and Makedon (2006) propose *k*-means-based reconstruction approach to recover *z*-score data from normalized masked ratings. Their algorithm tries to cluster *z*-scores into groups so that original ratings can be discovered. Calandrino et al. (2011) utilize auxiliary information to infer information about customers of CF systems and they test their attack with online websites. The authors devise *knn* attack for neighborhood-based CF systems. If the part of a user vector is known before, an attacker inserts k fake users identical to the disclosed user. When one of the k fake users asks for a prediction for items in place of the attacker, Calandrino et al. (2011) state that k-1 fake neighbors and the disclosed user will constitute k neighbors so that the prediction comes from the disclosed user. Huang and Du (2008) work with RRTs to discover optimal scheme. They quantify both

privacy and utility stating that considering one of them for an optimal scheme would be a bad decision. The approaches proposed in (Kargupta et al., 2003; Zhang, Ford and Makedon, 2006; Calandrino et al., 2011) target CF systems particularly. Kargupta et al. (2003) and Zhang, Ford and Makedon (2006) recover numerically rated schemes, and Calandrino et al. (2011) make inferences.

Studies given in this chapter are mostly related to numeric format and covers PPDM algorithms. However, tastes of users are not always expressed in numeric format in CF systems. Especially, if data is about the preference of an item, it can be coded in binary such as *like* or *dislike*. At this point, it is important to emphasize that the focus of this dissertation is to derive perturbed binary data because there is no concentrated effort to reconstruct perturbed binary data although Huang and Du (2008) quantify utility and privacy for RRTs. The work presented in this dissertation targets central, partitioned, distributed and P2P PPCF schemes to derive private data of users or data holders by exploiting auxiliary information to improve results or overcome bottlenecks.

1.5. Contribution

The problem focused on this dissertation is to derive original ratings of individuals or institutional data in binary PPCF systems. The literature in PPCF for binary rated data covers various schemes based on how data is stored. As mentioned, data could be stored centrally by a server or distributed between two- or multi-parties and even each user can collaborate for a P2P network to obviate the need for data holders. This dissertation targets PPCF schemes with these data partitioning scenarios. The contribution of this dissertation can be summarized as follows:

1. The first contribution of this dissertation is to develop an attack technique on the central server-based binary PPCF scheme (Polat and Du, 2006) to derive user ratings perturbed by RRT with multi-group. The authors of this targeted scheme utilize RRT, but this technique of disguising binary rating values allows a malicious data holder to estimate the ratings of any item by using θ value. This information is exploited to recover original binary ratings.
2. In addition to disguising original rating values of users, the scholars propose to insert some fake ratings into user vectors so that the data holder cannot distinguish the genuine ratings from the fake ones. In this dissertation,

auxiliary information is utilized to tackle the problem of fake ratings to discover genuine ratings.

3. Regarding distributed PPCF schemes, two attacks are proposed and two other attacks from the literature have been applied to P3CF and PPDCF schemes to derive institutional data by exploiting similarity values exchanged between parties. NBC-based horizontal PPCF schemes are immune to data disclosure attacks because the master party does not have the value of the queried item, q. This problem is overcome by introducing auxiliary information to estimate the value of q.
4. In terms of P2P PPCF, three attacks have been applied to disclose individual privacy. Auxiliary information is exploited to recover ratings of peers in order to eliminate bottlenecks due to horizontal nature of P2P collaboration.

1.6. The Data Set and Evaluation Criteria

MovieLens Million (MLM)[1] is a well-known, frequently used benchmark data set in CF and PPCF community and it has been used throughout all experiments in this dissertation. MLM is a movie rating data set on a discrete scale from 1 to 5 where 1 indicates the lowest preference and 5 represents the opposite. In MLM data set, there are 1,000,209 ratings associated with 6,040 users for 3,883 items (movies), which makes its density roughly 4.3%. MLM can be considered a sparse data set, which is very common for CF systems. One of the motivations of this dissertation is to utilize auxiliary information while deriving private individual or institutional data. In this context, auxiliary information needs to be collected about MLM data, and it is collected from Internet Movie Database[2] (IMDB) website.

This dissertation deals with binary rated PPCF schemes; however, MLM data contains numeric ratings. Thus, numeric scales are converted to their binary equivalences (Miyahara and Pazzani, 2000) to constitute a matrix of binary ratings. Ratings greater than 3 are converted to *like* and the rest other than unrated ones are converted to *dislike* for MLM data set. Details about the data set is given in Table 1.1.

[1] https://grouplens.org/datasets/movielens/1m/
[2] http://www.imdb.com

Table 1.1. *Details of the data set*

Data set	User x Items	Density	Rating Scale	Binary Conversion
MLM	6,040 x 3,883	4.26%	5 star	If greater than 3, marked as *like* Otherwise, marked as *dislike*

As evaluation criteria, precision (*prec*) and recall (*rec*) have been used. Since MLM and conventional data sets are highly sparse, conventional *prec* and *rec* calculations will be dominated by unrated entries. Therefore, *prec* and *rec* are calculated by considering on recovered rated items. *Prec* is useful to understand how much of the derived *like*s and *dislike*s were indeed identical to the original data. *Rec* gives the ratio of how much of the original *like*s and *dislike*s are recovered. An example of confusion matrices for the first and second aspect of privacy is given in Table 1.2. When an attack attempts to derive original ratings for a PPCF scheme, the outcoming confusion matrix is given in Table 1.2.a. If an attack attempts to derive whether an item is rated or not, the outcoming confusion matrix will become a 2 × 2 matrix as in Table1.2.b. After ratings are derived, and the confusion matrix is composed, *prec* and *rec* are computed by dividing the sum of correctly classified *like*s and *dislike*s to the sum of row and column values of *like*s and *dislike*s, respectively. The calculation of *prec* and *rec* are given in Eq. 1.1 for the first and the second aspect of privacy, respectively.

Table 1.2. *Confusion matrices*

a) *The first aspect of privacy*

		Original		
		Likes	**Dislikes**	**Unrated**
Classified	**Likes**	V_{11}	V_{12}	V_{13}
	Dislikes	V_{21}	V_{22}	V_{23}
	Unrated	V_{31}	V_{32}	V_{33}

b) *The second aspect of privacy*

		Original	
		Rated	**Unrated**
Classified	**Rated**	Z_{11}	Z_{12}
	Unrated	Z_{21}	Z_{22}

1.7. Organization

The rest of the dissertation is organized as follows. In Chapter 2, details about the targeted PPCF systems and preliminaries needed throughout the dissertation are given. In Chapter 3, proposed scenarios to derive private data from central-based binary PPCF schemes are given. In Chapter 4, two- and multi-party binary PPCF schemes are

investigated through different attack techniques to derive private institutional data. In Chapter 5, P2P binary PPCF schemes are scrutinized to derive private peer data. In Chapter 6, concluding remarks about the dissertation are discussed.

$$prec = \frac{\sum_{i=1}^{2} V_{ii}}{\sum_{i=1}^{2}\sum_{j=1}^{3} V_{ij}},\ rec = \frac{\sum_{i=1}^{2} V_{ii}}{\sum_{i=1}^{3}\sum_{j=1}^{2} V_{ij}} \qquad (1.1)$$

$$prec = \frac{Z_{11}}{Z_{11} + Z_{12}},\ rec = \frac{Z_{11}}{Z_{11} + Z_{21}}$$

2. PRELIMINARIES

In this chapter, preliminary information required throughout the dissertation is given. This dissertation is focused on deriving private user or institutional data from binary PPCF schemes with different data partitioning scenarios. Therefore, related data perturbation methods, prediction or recommendation protocols of binary PPCF schemes targeted in this dissertation will be introduced in the following subsections.

2.1. Randomized Response Technique – RRT

RRT is a survey method proposed by Warner (1965) to estimate the prevalence of a sensitive attribute in a population. The main purpose of RRT is to protect the privacy of respondents. The sensitive question is a polar one whose answer is either positive or negative. In an RRT setting, a threshold, θ, is determined in advance and each respondent is notified about θ before the sensitive question is released. Then, each respondent is asked to use a device that generates a random number from the range (0, 1] before responding the sensitive question. The sensitive question is asked and the respondent of the survey is asked to give an opposite answer to the question if the random device generates an output greater than θ. Otherwise, the respondent gives a true answer for the sensitive question. On the other hand, the interviewer does not know whether the respondent gives a true or opposite answer to the sensitive question. Since respondents utilize a random device, the percentage of the population who gives their true answers to the sensitive question is approximately θ while the percentage of the population giving their opposite answer is $1-\theta$. Assume that φ is the percentage of yes answers after the interviewer collects the answers. Let π be the percentage of the sensitive attribute in the population. φ can be formulated as seen in Eq. 2.1.

$$\varphi = \theta\pi + (1-\theta)(1-\pi) \tag{2.1}$$

π can be estimated after Eq. 2.1 is rewritten for it. Eq. 2.2 displays the estimation of π. Although the reviewer does not know which respondent gives a true answer, he or she can estimate the true percentage of the population with sensitive attribute without disclosing their privacy. Such a method can encourage people to express their true opinion on a sensitive issue that otherwise they hesitate to reveal.

$$\pi \approx \frac{\varphi + \theta - 1}{2\theta - 1} \tag{2.2}$$

RRT could be very useful for disguising binary data. Any binary rated data has two options either *like* and *dislike* if it is rated. Therefore, it is very convenient for users to perturb the data set by utilizing RRT based on predetermined or random θ. RRT deals with the first aspect of privacy, which disguises the actual rated values of items in the dataset.

2.2. Naïve Bayes Classifier - NBC

Bayes' rule of conditional probability gives the probability of an event to occur when the occurrence of another event is observed. Assume that the class membership probability of X will be calculated. It is denoted by $p(c_i \mid X)$ where c_i is a class variable and X's class membership for c_i will be calculated. This conditional probability is called *posterior probability* and defines the conditional relationship between c_i and X. Bayes' rule states that the posterior probability can be calculated by using probabilities related to it. Eq. 2.3 gives the Bayes' rule. The equation states that the posterior probability of $p(c_i \mid X)$ can be calculated by utilizing the posterior probability of $p(X \mid c_i)$, the prior probability of event $p(X)$ and $p(c_i)$.

$$p(c_i \mid X) = \frac{p(X \mid c_i)\, p(c_i)}{p(X)} \tag{2.3}$$

Given X has *m* features, one has to calculate $p(x_1 \mid c_i)$, $p(x_2 \mid c_i)$,…, $p(x_m \mid c_i)$ in order to obtain $p(X \mid c_i)$, and the naïve assumption requires that these features be independent. Eq. 2.3 is calculated for each class and X is assigned to the class with the highest probability. $p(X)$ can be ignored because it will yield the same result every time. Since the naïve assumption is that features are independent from each other, $p(X \mid c_i)$ can be expressed as seen in Eq. 2.4 (Han, Kamber and Pei, 2012). Polat and Du (2006), Kaleli and Polat (2007a; 2007b; 2010; 2015) utilize NBC while producing private recommendations. Details about these schemes will be given later in this chapter.

$$p(X \mid c_i) = \prod_{k=1}^{m} p(x_k \mid c_i) \tag{2.4}$$

2.3. Methods for Hiding Rated Items

Beside disguising the actual rating values, a malicious server or party should not identify which items are indeed rated, which is the second aspect of privacy. Therefore, another data perturbation phase is needed in addition to perturbing the actual rating values. The authors of the studies targeted in this dissertation (Polat and Du, 2005c, 2006, 2008; Kaleli and Polat, 2007a, 2010, 2015) usually employ various methods of inserting some fake ratings into the original vector so that rated items cannot be differentiated from unrated fake items. This dissertation handles PPCF schemes with central server-based (Polat and Du, 2006), P3CF (Polat and Du, 2005c, 2008; Kaleli and Polat, 2007a), PPDCF (Kaleli and Polat, 2015) and P2P (Kaleli and Polat, 2015) schemes and the method of hiding rated items in these schemes are given in this part.

The central server-based binary PPCF schemes by Polat and Du (2006) fills unrated items by checking the number of genuine rated items in the original user vector. A user, u, finds the number of rated items in his or her vector, m_u, and picks a randomly drawn uniform number, m_{up}, from the range (1, m_u). Then, u selects m_{up} items from unrated items and fills half of them with *like* and the other half with *dislike*. The density of a user vector, d, can be defined as the number of rated items divided by the number of all items in the original user vector. Then, this method of filling unrated items can be associated with d. Any u can fill his or her rating vector with fake items as much as d in the worst-case.

Polat and Du (2005c; 2008) propose HPD- and VPD-based binary P3CF schemes. The authors discuss that their schemes can be extended to PPDCF schemes as well. In these schemes, institutional privacy is handled because users send their original data to a data holder (server) without perturbing their vectors. Servers (data holders) must maintain privacy for their institutional data to prevent any disclosure. The method of hiding rated items proposed for by Polat and Du (2005c; 2008) is called *private similarity computation protocol* (PSCP). In PSCP, each party finds the number of rated items, m_u, for each user they have. Then, they compare m_u with m, the number of total items. If $m_u \geq m/2$, then the party picks a random number drawn uniformly from the range (1, m) and randomly removes the corresponding number of rated items from the user vector. If $m_u < m/2$, the party picks a uniform random number from the range (1, $m-m_u$) and fills randomly selected unrated cells with the default votes of corresponding items that are calculated privately. Kaleli and Polat (2007a) apply a similar method for the NBC-based VPD binary P3CF scheme. The non-master party picks a random number drawn uniformly over the

range [1,100] and fills that percentage of unrated items of AU's vector with default votes. The filling methods proposed by Polat and Du (2005c; 2008) and Kaleli and Polat (2007a) fills the user vector with a random number of unrated items. If the number of appended ratings is proportionally very large compared to m_u, then the output user vector will become much more different from the original rating. However, the idea in PPCF is to balance privacy and accuracy. Dramatically perturbed data for the sake of privacy would affect the prediction accuracy. In another NBC-based HDD and VDD binary PDPCF scheme by Kaleli and Polat (2015), the authors associate the number of fake items to be appended into the user vector with d to alleviate such a downside. The party having AU initiates a prediction process and determines a filling factor, δ_{AU}, where δ_{AU} can be factors of d such as 0.25d, 0.5d, d, 2d or 4d. The related party with AU randomly pick a uniform number, L, from the range (1, δ_{AU}]. There might occur some cases in sparse data sets where δ_{AU} is less than 1, so the range can be updated as (0, δ_{AU}]. Then, L percent of unrated items of AU are filled with either *default vote*s (DV) or *random filling* (RF) method. A default vote for an item is the dominant rating in a vector. In other words, if *like*s are more than *dislike*s in a user vector, then *like* is the default vote for that user. In RF, a random binary value inserted into each selected unrated item cell, independently. This method of hiding rated item cells by filling unrated items based on δ_{AU} is called *hiding rated items* (HRI) protocol. An illustration of HRI is given in Figure 2.1. Filled unrated items are shaded. When DV is utilized, the dominant rating, which is *like*, is appended into the selected unrated items' cells. When RF is utilized, selected unrated items' cells are appended with random values.

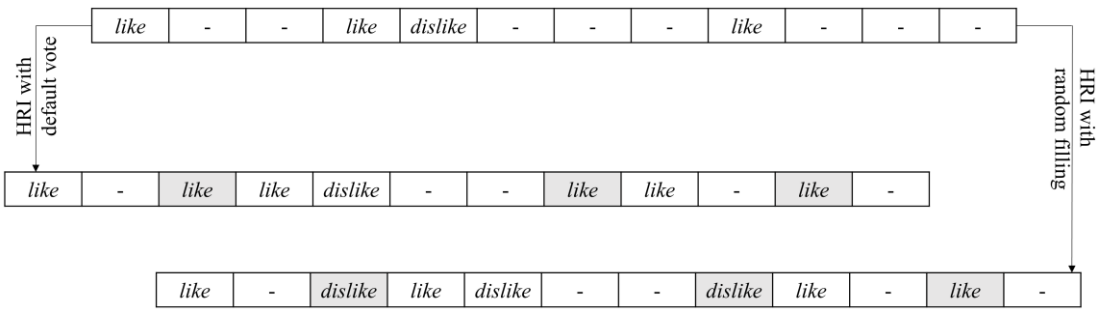


Figure 2.1. *An illustration of the HRI protocol*

In the P2P binary PPCF scheme (Kaleli and Polat, 2010), peers employ a very similar method to PSCP. In this setting, active peer, AP, picks a random density value, α_{AP}, between 0 and 100. Then, they pick another random integer δ_{AP} from the range [0,

α_{AP}]. Peers fill half of the δ_{AP} percent of unrated cells with *like*s and the remaining half with *dislike*s.

Filling methods other than HRI in distributed schemes including P2P could result in a user vector filled with fake appended ratings. Therefore, non-central server-based schemes will utilize HRI protocol to fill unrated items of user vectors. HRI protocol associates the size of fake ratings to be inserted with the factors of *d*. On the other hand, the central server-based PPCF scheme is exempted from this generalization because it fills unrated items up to *d*.

2.4. Central server-based Binary PPCF

The central server-based binary PPCF scheme targeted in this dissertation is proposed by Polat and Du (2006). The authors devise a PPCF scheme to offer recommendations by utilizing RRT as a data disguising method. RRT can be easily adapted into PPCF to perturb binary data. Each user, *u*, selects a random value r_u uniformly randomly over the range [0, 1] and checks it against a predetermined value of θ. If $r_u < \theta$, then the user sends the true vector to the server as is. Otherwise, he or she sends the false rating vector (the exact opposite of the rating vector). A false rating vector is created by applying *not* operation on binary ratings. Assume that V_u = (0110-1-0-0) is a binary rated vector of a user, *u*, where 1, 0 and - means *like*, *dislike* and an unrated item, respectively. If randomly generated uniform number r_u is less than θ, then the user sends V_u as it is. Otherwise, binary rated items in V_u are reversed, and *u* sends V_u = (1001-0-1-1). A couple of scenarios can be used to determine θ value. In the *constant* θ case, both the server and users agree on a predetermined θ value. In the *random* θ case, each user randomly picks θ value selected uniformly randomly from the range [0, 1] and the server is not aware of it. Notice that larger θ values mean that users' rating vectors will be preserved in most cases. For example, the server can expect that 80 percent of incoming data from the users are preserved, when θ is 0.8 in the constant θ case.

Although users apply RRT before sending their data to the server, the server can disclose user vector if it knows a true rating for any item. In such a case, the server can understand whether the user vector is reversed or preserved by simply checking the rating of the item whose true value is known by the server in perturbed data. If the related rating in the perturbed vector is equal to the true value of it (known by the server), then the server infers that the user preserves the rating vector; otherwise, the rating vector is

reversed. As a result, whole rating vector of a user is disclosed when a rating is known by the server. However, users might alleviate the effects of such an occurrence if rating vector is divided in multi-groups. Polat and Du (2006) propose that each user can divide their rating vector into G groups, where $1 \leq G \leq m$ and RRT can be applied to each group independently by users. Figure 2.2 displays RRT with 1-, 2- and 5-group. In the figure, groups are split by the color tones to differentiate from each other visually and the relation between sample r_u and θ values for each group is given just above each group. Suppose that a malicious server knows the true value of the first item of the related user vector, it discloses all user rating vector in 1-group while it can disclose the first half of the user vector in 2-group scheme. On the other hand, the server can only disclose the 20 percent of the original user vector in 5-group scheme. As a result, items that are in the same group with the item whose rating is known by the malicious server are disclosed. Items in other groups remain private in case of such disclosure. Therefore, the multi-group approach is another level of data disguising accompanied with RRT.

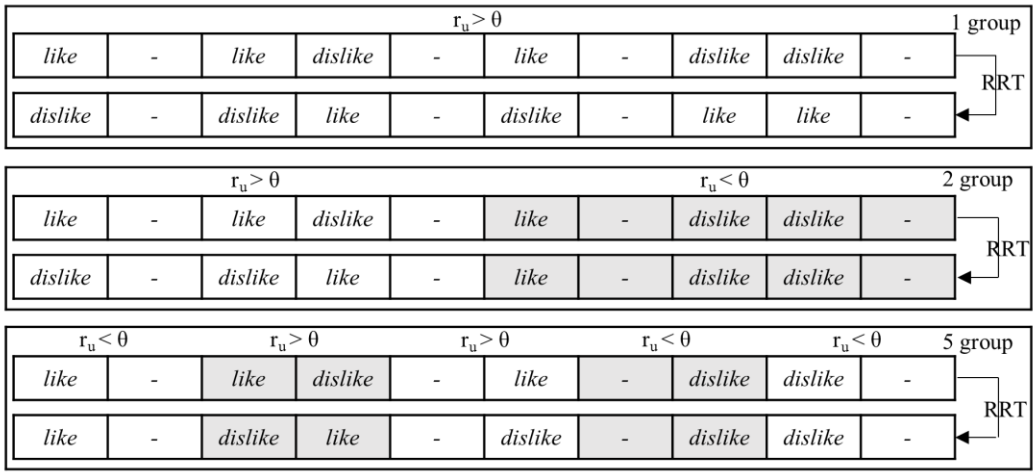


Figure 2.2. *RRT with multi-group*

In terms of the second aspect of privacy, the genuinely rated items are disguised by filling unrated items. The idea is to confuse the data holder about genuine ratings. As discussed in Chapter 2.3, users fill their unrated items with half *like*s and *dislike*s with a random density drawn from (0, *d*]. Therefore, the maximum number of unrated items to be filled is limited to the number of rated items in the original user vector.

Polat and Du (2006) hypothesize that recommendation can be made if they can reconstruct the original rating. They utilize Bayes' theorem to recover original matrix and produce recommendations. Given that Y is perturbed version of the original data, X, by

RRT and the server has to find a way to reach X. The authors utilize Bayes' theorem to reach X from Y, $p(X \mid Y)$, as given in Eq. 2.5.

$$p(\mathrm{X} \mid \mathrm{Y}) = \frac{p(Y \mid X)\, p(X)}{p(Y)} \tag{2.5}$$

The data holder knows that $p(Y \mid X)$ is θ owing to RRT. $p(Y)$ can be calculated from the perturbed data (Polat and Du, 2006). $p(X)$ can be calculated by utilizing θ as calculated in Eq. 2.1 and 2.2. When Eq. 2.5 is rewritten, Eq. 2.6 is obtained as follows.

$$p(X \mid Y) = \frac{\theta^2 + \theta p(Y) - \theta}{2\theta p(Y) - p(Y)} \tag{2.6}$$

2.5. HPD- VPD-based Binary P3CF Schemes

Two different data holders can decide to collaborate to enhance their data matrix for better recommendations. The basic assumption with distributed PPCF schemes is that parties hold original user vector and they should preserve institutional privacy while interacting each other. HPD- and VPD-based binary P3CF schemes targeted to obtain the private original institutional data in this dissertation are proposed by Polat and Du (2005c; 2008) and Kaleli and Polat (2007a).

P3CF schemes proposed by Polat and Du (2005c; 2008) are recommendation based algorithms, which returns top-*N* recommendations. In these schemes, a party associated with AU or an AU sends an initial query along with N_{AU} items, among which the AU wants the top-*N* recommendations to be produced. The number of items in $\mathrm{N_{AU}}$ is limited to $\mathrm{N} < \mathrm{N_{AU}} < m\text{-M}$, where M is the number of rated items of the AU. The authors employ a similarity metric based on the difference of the number of similarly, $t(s)$, and dissimilarly, $t(d)$, rated items over commonly rated items, $t(c)$, as shown in Eq. 2.7. The scholars offer two different HPD-based P3CF schemes based on how neighbors are selected. One of them selects best-*k* neighbors while the other one picks neighbors with a higher similarity with AU than a predefined threshold (τ), the threshold scheme. After neighbors are determined, the recommendation algorithm performs a column-wise sum of the number of *likes*, l_i and *dislikes*, d_i for each item. If $l_i - d_i > 0$, then the related item will be liked by the AU. Items whose $l_i - d_i$ results are greater than 0 are sorted in descending order, and top-*N* items will be the list of recommendation.

$$w_u = \frac{t(s) - t(d)}{t(c)} \tag{2.7}$$

Assume that there are two parties, A and B. The threshold scheme is given below. As discussed, this scheme utilizes τ to decide which users will become neighbors.

1. AU sends his or her ratings to A and B together with N_{AU} items.
2. The party A selects neighbors whose similarity is higher than τ. Then, it might add a random number over the range [−α, α] to τ to avoid B from making any inference about A's institutional data.
3. Then, A calculates $ld_{Ai} = l_i - d_i$ for $i = \{1, 2, \ldots, N_{AU}\}$ and sends it to B.
4. B selects best neighbors and calculates ld_{Bi}. B adds ld_{Bi} to ld_{Ai}, which is received from A. Then, the summation is sorted in descending order, and top-N items among N_{AU} are returned as recommendations to AU.

The below is the protocol for best-*k* where neighbors are sorted and best *k* of them are selected.

1. AU sends his or her rating along with the query to A and B.
2. The party A calculates similarities by applying PSCP on AU's rating vector. Then, A permutes and take absolute values of similarities and sends the similarities to B.
3. B calculates its own similarities and finds the best-*k* neighbors. Then, B calculates ld_{Bi} values. B sends ld_{Bi} values and best neighbors of A among best-*k* neighbors.
4. The party A identifies its own best neighbors by reverting permutation process and adds ld_{Ai} to ld_{Bi}. Then, the summation is sorted and the top-N list is returned.

Beside HPD-based schemes, Polat and Du (2008) propose VPD-based binary P3CF schemes as well. The authors consider two cases for VPD-based P3CF schemes. The list of N_{AU} items where recommendations will be produced could be either held by one of the parties, which is called Case-All, or shared between two party, which is called Case-Split. The details of Case-All are given below in steps. Notice that B acts as the master party.

1. AU splits his or her rating vector into two for A and B. AU sends corresponding parts of the vector to A and B. The list of N_{AU} items are sent to B since B holds those items.
2. The party A finds the partial similarity values by applying PSCP because A has part of the ratings. For each user, A sends these partial similarity values to B.
3. B also finds its own partial similarity values and calculates final similarity values by adding the ones received from A. Then, B selects best-*k* neighbors. However, B adds some random numbers drawn from the range $[-\alpha, \alpha]$ and $[-\gamma, \gamma]$ to τ and *k*, respectively, to prevent A from disclosing any further information.
4. After selecting the neighbors, B calculates ld_{Bi} and sorts it in descending order and returns top-N list. B does not need ld_{Aj} values because all of N_{AU} items reside on B's side.

The Case-Split case is very similar to Case-All, the only difference is that N_{AU} items are split between two-party. Therefore, $ld_j = ld_{Ai} + ld_{Bi}$ is calculated collaboratively as below.

1. AU splits his or her rating vector into two for A and B. AU sends corresponding parts of the vector to A and B.
2. B calculates partial similarity values by utilizing PSCP and sends them to A.
3. The party A calculates its own partial similarity values and adds them to the partial similarity values received from B in order to calculate the final similarities for each user. Then, A selects neighbors with random τ and sends neighbors to B and sign of the similarities associated with neighbors.
4. B calculates ld_{Bi} values for the neighbors and sends them to A.
5. The party A must calculate ld_{Ai} values; however, A reselects own neighbors to prevent B from disclosing any information after the recommendation. Therefore, it applies random τ and *k* to select its neighbors. Then, it calculates ld_{Ai} values and adds them to ld_{Bi} to return top-N list.

The HPD- and VPD-based binary P3CF schemes by Kaleli and Polat (2007a) is an NBC-based prediction scheme contrary to the top-N recommendation introduced by Polat and Du (2005c; 2008) as discussed until now in this part. Miyahara and Pazzini (2000) employ NBC for CF purposes on binary data. In NBC-based CF, each user is a feature, and corresponding ratings are feature values. Kaleli and Polat (2007a) also employ NBC for prediction purposes on partitioned data. Given that an item has *n* features and the probability of this item to belong a class, c_j where $j \in \{like, dislike\}$ can be expressed as given in Eq. 2.8 (Kaleli and Polat, 2007a).

$$p\left(c_j \mid f_1, f_2, ..., f_n\right) = p\left(c_j\right)\prod_{i}^{n} p\left(f_i \mid c_j\right) \tag{2.8}$$

Eq. 2.8 needs to be rewritten when data is horizontally partitioned because each party will have half of the features. Assuming that n_f is the number of users that the first party has and Eq. 2.8 can be expressed as:

$$p\left(c_j \mid f_1, f_2, ..., f_n\right) = p\left(c_j\right)\prod_{i=1}^{n_f} p\left(f_i \mid c_j\right)\prod_{i=n_f+1}^{n} p\left(f_i \mid c_j\right) \tag{2.9}$$

Eq.2.9 displays that *n* feature of an item is partitioned between parties; therefore, each party must calculate its part, and the interim results must be then multiplied. The authors give the related HPD-based scheme presuming that B is a master party as follows:

1. *AU* send his or her data both parties and B calculates $p(c_j)$.
2. Each party computes own part of conditional probabilities. A sends its own conditional probabilities to B.
3. B calculates the final probability value once A's interim conditional probabilities are received.

Notice that there is no data hiding method is employed in the NBC HPD-based P3CF scheme. Since both of the parties exchange aggregated values of conditional probabilities, the scholars claim that the scheme preserves the privacy of institutional data.

In HPD-based binary P3CF (Kaleli and Polat, 2007a), probabilities can be calculated easily; the VPD-based scheme requires the exchange of interim results to reach the final conditional probability because *q* is held by one of the parties. Since

corresponding parts of AU's query vector is received by each party, parties should calculate their own nominator, PN, and denominator, PD, to compute $p(f_i \mid c_j)$ as shown in Eq. 2.10, where PN_1 and PN_2 show the nominator values of the first and the second party, respectively. Likewise, PD_1 and PD_2 show the denominator values of each party. VPD-based scheme with B being the master party is defined as follows:

1. AU sends corresponding parts of the his or her rating vector to A and B. AU also calculates $p(c_j)$ and sends it to B along with the vector.
2. Party A calculates own part of the conditional probability by hiding rated items of AU to prevent B from disclosing information. The party A employs the related filling method covered in Chapter 2.3, which does not take *d* into account. Then, A calculates PN_1 and PD_1 and sends them to B.
3. B adds PN_1 and PD_1 received from A to PN_2 and PD_2, which are calculated by B, respectively. B now can calculate the final conditional probability.

$$p\left(f_i \mid c_j\right) = \frac{PN_1 + PN_2}{PD_1 + PD_2} \tag{2.10}$$

2.6. HDD- and VDD-based Binary PPDCF Schemes

Beside P3CF schemes, where two data holders collaborate, many data holders can also collaborate to produce CF recommendations with privacy. The main objective of the parties collaborating in PPDCF is also to preserve the privacy of institutional data. Similar to P3CF schemes, users send their original data and parties perform data exchange with privacy. Polat and Du (2008) explain how their P3CF schemes (Polat and Du, 2005c, 2008) can be extended to PPDCF. Kaleli and Polat (2015) extend their P3CF schemes (2007a) to PPDCF in their study as well.

Polat and Du (2008) discuss in their paper that HPD- and VPD-based binary P3CF schemes can be extended to PPDCF with slight modifications. To adapt the threshold based HPD scheme into PPDCF, each party picks its own best neighbors and calculates ld_{lj} where subscript *l* varies between 1 to the number of parties. Then, these values are sent to the selected master party for the top-N recommendation. In the best-*k* HPD-based binary P3CF scheme, the parties calculate similarities and send them to the master party for the best neighbor selection. After the best neighbors are selected, the master party lets each party know about their users who manage to get into the neighborhood. Each party

calculates ld_{lj} values and sends it to the master party for the top-N recommendation. Case-All and Case-Split, VPD-based binary P3CF schemes, can be adapted to PPDCF as well. In Case-All, each party calculates their own similarity values and lets the master party know them. Since the master party has all N_{AU} items, the master party adds the values received from parties to the ones calculated by itself. After neighbors are selected, the master party calculates ld_{lj} and returns top-N recommendation. In Case-Split scenario, similarities are calculated and sent to the master party for neighborhood selection. The master party selects neighbors and broadcasts it to the parties that own an item in N_{AU} list. Then, these parties calculate ld_{lj} values and send them to the master for the top-N recommendation.

Polat and Du (2008) only discuss that HPD- and VPD-based binary P3CF schemes can be extended in HDD- and VDD-based manner for binary PPDCF schemes as cited above. However, they do not include the extended PPDCF versions in their experiments. P3CF schemes by Polat and Du (2005c; 2008) are extended to PPDCF in this dissertation in Chapter 4 to derive private institutional data. Because P3CF can be extended to PPDCF schemes, they are only discussed Chapter 4.7 in the experiments, where the number of parties participated in PPCF process are discussed.

NBC-based HDD and VDD binary PPDCF schemes are proposed by Kaleli and Polat (2015). These schemes provide a prediction for q contrary to recommendation provided in the HDD- and VDD-based binary PPDCF schemes (Polat and Du, 2008). The NBC-based PPDCF schemes differ its P3CF version by the way AU's vector is perturbed. The authors apply HRI and RRT with multi groups on AU's data. Since AU's rating vector is divided into multi-groups, the HDD- and VDD-based binary PPDCF schemes become more complicated compared with the HPD- and VPD-based binary P3CF schemes given in the previous subsection Chapter 2.5. A master party is determined before the NBC-based HDD and VDD binary PPDCF schemes start. Therefore, the AU sends her ratings and q to the master party. The first thing that the master party does is to perturb AU's data by HRI, which fills unrated items based on d, vector density. After HRI is applied, the master party applies RRT with multi-group. However, the master party applies multi-group RRT in a different way. For each group, the master party determines a random θ over the range (0, 1] rather than a fixed predetermined one as applied in the central server-based PPCF scheme.

In the HDD-based binary PPDCF scheme by Kaleli and Polat (2015), it is trivial that Eq. 2.9 must be extended to each party. Each party must be able to calculate the conditional probability for their part without interacting with other parties. However, AU's data is perturbed with random θ and the collaborating parties do not know whether the rating vector of AU is true or false due to RRT with multi-group. Only the master party knows it. Therefore, the conditional probabilities for each feature in each party must be calculated using Eq. 2.11. PN_{ig} denotes the nominator value for *i*-th feature, where *i* can range between 1 and the number of users the relevant party has, for the *g*-th group, where $g = \{1, 2, \ldots, G\}$. Likewise, PD_{ig} denotes the denominator value for the *i*-th feature and *g*-th group. Although each party uses Eq. 2.11 for the conditional probabilities, they should calculate it twice for both $j = like$ and $j = dislike$ because of RRT. The protocol for HDD-based binary PPDCF scheme is given below.

1. AU' s rating vector with *q* is received by the master party. The master party appends AU' s vector with HRI and perturbs it RRT with multi-groups. The perturbed vector is sent to other parties.
2. Each party calculates PN_{ig} and PD_{ig} values for each feature they have and group *g* for $j = like$ and $j = dislike$. Then, each party sends these interim results to the master party.
3. Once all interim results for the conditional probabilities received from other parties, the master party picks the right $p(f_i \mid c_j)$. Then, the final result is calculated, and prediction about *q* is returned.

$$p\left(f_i \mid c_j\right) = \frac{\sum_{g=1}^{G} PN_{ig}}{\sum_{g=1}^{G} PD_{ig}} \tag{2.11}$$

In the NBC-based VDD binary scheme by by Kaleli and Polat (2015), conditional probabilities can be calculated by utilizing Eq. 2.11 as well. Due to multi-groups, each party calculates $p(f_i \mid c_j)$ for both $j = like$ and $j = dislike$ in a similar way to NBC HDD-based scheme. However, in vertically shared data, *q* is held by the master party. Therefore, collaborating parties have to calculate $p(f_i \mid c_j)$ for both $f_i = like$ and $f_i = dislike$ contrary to the NBC-based HDD scheme; fortunately, $p(f_i = like \mid c_j) + p(f_i = dislike \mid c_j) = 1$. The NBC-based VDD binary scheme is described as follows.

1. AU sends her rating vector and *q* to the party having *q*, which becomes the master party. The mater party perturbs AU' s rating vector with HRI and RRT with multi-groups. Then, the master party sends the corresponding parts of the rating vector to the relevant parties.
2. Each party calculates PN_{ig} and PD_{ig} values for each feature they have and group for *j = like* and *j = dislike*. Then, each party sends these interim results to the master party.
3. The master party checks *q* and picks the correct conditional probability for *j = like* and *j = dislike*. Then, the final conditional probability is calculated, and the prediction about *q* is returned.

2.7. P2P Binary PPCF schemes

P2P binary PPCF scheme (Kaleli and Polat, 2010) is an NBC-based scheme whose foundation is similar to the NBC-based P3CF (Kaleli and Polat, 2007a) and PPDCF (Kaleli and Polat, 2015) schemes. In this setting, users (peers) collaborate with each other to establish a CF system with privacy instead of relying on data holders. Since each peer participates in the prediction process individually, they should put efforts to preserve their privacy while interacting with other peers so that individual privacy of peers can be maintained. In P2P setting by Kaleli and Polat (2010), AP perturbs her data for each peer separately. AP applies RRT with multi-groups and fills unrated items with random density. AP picks a random θ_i and G_i values for each peer where $i = \{1,2, \ldots, n\}$. θ_i are selected from the range [0, 0.5] while G_i is selected from the range between [2, γ], where $\gamma \geq 2$. However, Kaleli and Polat (2010) stress that a large value of γ could cause performance issues. AP utilizes a method that fills the unrated items with a random density. As discussed in Chapter 2.3, a query vector filled with random density could alter the vector dramatically. Therefore, HRI protocol, which associates *d* with the number of fake items to be inserted into unrated items' cells, is preferred in this dissertation instead of filling peer's unrated items with random density. AP applies RRT with multi-group and HRI protocol on her vector for each peer. Therefore, a different copy of the rating vector is sent to each different peer. In terms of the calculation of the conditional probability, each peer can compute it by utilizing Eq. 3.11 once they receive the AP's vector and *q*. This scheme is similar to the NBC-based PPDCF (Kaleli and Polat, 2015) because each peer has *q*. Thus, AP is not aware of whether $p(f_i \mid c_j)$ is calculated for *fi* =

like or fi = dislike. Still, peers have to calculate PN_{ig} and PD_{ig} values where $g = \{1, 2, \ldots, G_i\}$ because of RRT with multi-group. The details of the scheme are described below.

1. AP broadcast a participation request. Peers who would like to join in the prediction process return a positive answer.
2. AP perturbs her rating vector with RRT with multi-groups and sends the perturbed rating vector with q to peers who wish to participate in the prediction process.
3. Each peer having q computes PN_{ig} and PD_{ig} required for the conditional probability calculation for each group.
4. AP picks the right PN and PD values and calculates the final conditional probability value. Then, the prediction about q is made.

In NBC-based PPDCF schemes (Kaleli and Polat, 2015), the master party adapts privacy measures to protect AU's data. The master party perturbs the user vector by employing RRT and HRI protocols. Nevertheless, the authors discuss that there might be cases where the master party can infer confidential information about other parties. The authors list two extreme cases threatening the NBC-based PPDCF schemes (Kaleli and Polat, 2015).

1. The similarity between AU and a user can be calculated only if the relevant user has a rating for q. Therefore, the master party infers that a user of a collaborating party did not rate q if the similarity value is not calculated for that user. In such a scenario, the master party can create a mapping of all users who rate q.
2. A user might have a rating for q but she might not have any commonly rated item with AU. The master party can detect such an incident when it receives the similarity values. This incident reveals that the relevant user did not rate any rated items in the AU's vector.

The first extreme case is not a threat for VDD because parties do not have q and must calculate interim results. Kaleli and Polat (2015) propose that collaborating parties in HDD-based schemes can circumvent the first extreme case by performing a modification on HRI protocol. The relevant party determines a random L value based on data set density, d_{set}, in place of d, which is the density of a user vector. Then, the relevant

party constructs a list of users who did not rate *q*. L percent of those users' *q* is filled with random or default value of the user vector. So, the master party could not differentiate who rated *q* because some randomly introduced users who did not rate *q* calculates similarity values.

The second extreme case is a concern for HDD- and VDD-based schemes. When the second extreme case is experienced, it means that a particular user has no commonly rated item with AU. A party with such users who experience the second extreme case applies HRI protocol on the unrated items that correspond the rated items of the AU. The new perturbed vector of those users would have some commonly rated items after this process. To fill unrated items, the relevant party fills L percent of corresponding items in the relevant user vector with default votes of the user.

In the NBC P2P binary PPCF scheme (Kaleli and Polat, 2010), similar extreme cases are addressed by the authors. The first extreme case is the exact same incident with the NBC-based HDD (Kaleli and Polat, 2015) scheme about discovering the users who rate *q*. If a peer does not rate *q*, she cannot join the prediction process. Therefore, peers who rate *q* is revealed by monitoring peers in the prediction processes for different *q*'s. The authors propose an extra privacy measure for peers to avoid such an incident. Each peer in the P2P network generates two uniformly random values λ_i and r_i between [0, 1]. If r_i is less than λ_i, the related peer should participate in the prediction process. Therefore, half of the peers join the prediction process regardless of the presence of *q*, rated or not. The second extreme case that the authors mention is a measure against an attack known as *acting as an active user*. This attack will be covered in Chapter 4; it is an effort to derive the ratings of users by manipulating a single cell every time a new query is dispatched by an AP or AU. The authors indicate that such an attack could be alleviated if peers apply a similar data hiding method to AP. Each time AP asks for a prediction, peers should apply HRI protocol for their vector. Thus, peers respond each query with a different rating vector.

To sum up, the remedies offered by Kaleli and Polat (2010; 2015) for extreme cases have similar foundations in PPDCF and P2P PPCF. When the first extreme case is encountered, the solution is to intervene in users' or peers' participation in PPCF process. Therefore, the measures taken against the first extreme case will be hereafter called *peer/party privacy for participation* (PPP) for convenience. On the other hand, the solution to prevent from the second extreme case is to perturb user or peer ratings.

Likewise, the measures taken against the second extreme case will be hereafter called *peer/party privacy for ratings* (PPR) for convenience.

3. DERIVING PRIVATE DATA FROM CENTRAL SERVER BASED BINARY PPCF SCHEMES

In this chapter, the central server based PPCF scheme by Polat and Du (2006) is scrutinized regarding the degree of privacy they provided. The authors claim that individual privacy can be kept by RRT proposed by Warner (1965) with multi-groups and a filling method of unrated items. Each user in these schemes perturbs own rating vector before submitting it to a central server to protect their individual privacy. The PPCF scheme handles the privacy of users in two aspects. The first aspect of privacy deals with the actual rating values made by users, and it is controlled by RRT with multi-groups. On the other hand, the second aspect of privacy aims to hide the information of whether an item is rated or not. This ascpect of privacy is maintained by inserting fake ratings into the original data so that the server does not distinguish the genuine rated items from the fake items.

The main concentration of this chapter is to reconstruct the original user vectors from the perturbed data so that user ratings can be disclosed. Since each user perturbs his or her own rating vector, the server has a perturbed data matrix collected from users. This chapter devises and describes attack techniques to reconstruct the original data from the perturbed data assuming that there is a malicious server whose intention is to derive confidential user data. Deriving the original data from the perturbed data has two goals. The first goal assumes that users mask their data with RRT without applying the filling method. Therefore, the first goal of the reconstruction is to derive rating values made by users. The second goal is to identify which items are indeed rated. This goal is related to the data hiding phase. There are two subsections that deal with the problem of data reconstruction in terms of the first and second aspect of privacy.

3.1. Reconstructing Actual Ratings

The motivation here is to derive actual rating values of users by an effort to reconstruct the original data matrix from the perturbed one. As discussed, RRT is utilized to disguise the ratings. The main idea in this data reconstruction attack is to discover the behavior of some critical items. Identifying the behavior of some particular items from the data set perturbed by users can help the malicious server reconstruct the original binary ratings of users. Items having special patterns such as mostly being rated *like* or *dislike* by users might be beneficial to anticipate potential ratings. Therefore, the first and

main goal is to extract *extreme items* that can be described as the ones that are rated either *like* or *dislike* by the overwhelming majority of users. Such items will be useful for deriving binary ratings from the perturbed data. Extreme items might reveal a lot about a data set even though RRT disguises ratings. For example, assume that a malicious server knows that it_1, it_2, and it_3 are extreme items, which are rated as *like* by most of the users. If the malicious server realizes that all of these three extreme items are rated as *dislike* in the disguised user-item matrix, it is highly possible that the related user has reversed his or her ratings before sending it to the server. Due to the nature of RRT, all items in the same group with extreme items can be disclosed.

The base method to reconstruct original binary ratings relies on determining extreme items. Extreme items can be easily recognized from an unperturbed (original) data set by calculating the column-wise sum of all items for the presence of *like* and *dislike*. Let $s_{i\text{-}like}$ and $s_{i\text{-}dislike}$ be the number of presence of *like* and *dislike* for *i*-th item, respectively. If either $s_{i\text{-}like}$ or $s_{i\text{-}dislike}$ is very large while the other is very low, then one can infer the rating that most of the users prefer to rate for the related item. The calculation of $s_{i\text{-}like}$ and $s_{i\text{-}dislike}$ for each item would not be achieved by simply doing column-wise sum when RRT is employed. Therefore, estimation of the prevalence of each item in order to figure out extreme ones requires some processing. Recall that Warner (1965) originally designed RRT to find out the percentage of a population belonging to a sensitive group. This condition is quite similar to extracting extreme items since the original data set is perturbed by RRT. Warner's (1965) simple assumption estimates the percentage of population having answered positive to a sensitive question. In this scenario, item ratings, whose values are binary, are perturbed. Therefore, one who wants to estimate the true percentages of items that are originally rated *like* or *dislike* must repeat the same calculations for each item. After RRT is applied, the estimated percentage of the population who rates *like* for *i*-th item can be calculated using Eq. 3.1 as follows:

$$p\left(X_i = like\right) = \pi\theta + \left(1-\pi\right)\left(1-\theta\right) \tag{3.1}$$

In Eq. 3.1, π is the true percentage of users rating the related item, X_i, as *like* before RRT. It is the value that needs to be estimated. Therefore, Eq. 3.1 states that the percentage of users in the perturbed data set whose rating for the related item, X_i, is *like* consists of the percentage of users who indeed rate it *like* (π) with probability θ and the percentage of users who indeed rate it *dislike* ($1 - \pi$) with the probability of ($1 - \theta$). If Eq.

3.1 is solved for π, an estimate about π (referred to as $\tilde{\pi}$) is obtained in terms of $p(X_i = 1)$ and θ as follows:

$$\pi = \frac{p\left(X_i = like\right) + \theta - 1}{2\theta - 1} \tag{3.2}$$

The notion expressed in Eq. 3.2 is used to estimate the true percentage of user population who rates a particular item *like* in the user-item matrix. Hence, it is trivial that the percentage of population of users rating the same item *dislike* is $1 - \tilde{\pi}$. The probability of an item being *like* or *dislike* will be called as *popularity index* or *unpopularity index*, respectively. Items with high popularity index values are the items with low unpopularity index and vice versa because the sum of popularity and unpopularity indexes of an item is complementary and equal to 1. Therefore, items with high popularity indexes can be considered as rated *like* and the ones with high unpopularity index values can be considered as rated *dislike* in the reconstruction. To determine extreme items, items in the perturbed data set disguised by RRT are sorted in descending order based on their popularity and unpopularity indexes after their popularity and unpopularity indexes are calculated using Eq. 3.2. Items with higher popularity and unpopularity indexes place themselves up in the list. The first *N* items are chosen as extreme items. *N* can be any number from the range [1, *m*]. Remember that *m* is the number of items. After extreme items are identified, the ones with high popularity indexes are marked *like* because they are more likely to be rated *like*. Similar process is done for extreme items with high unpopularity indexes by marking them with *dislike*. After extreme items are extracted from the perturbed data set, the reconstruction method checks if users rate the extreme items in the expected pattern. If an extreme item is marked as *like* (high popularity index), a user is expected to rate that extreme item *like*. On the contrary, an extreme item marked as *dislike* (high unpopularity index) is expected to be rated as *dislike* by users. Assume that there are twenty extreme items extracted from a data set, twelve of them are marked as *like* and eight of them are marked as *dislike*. If a specific user rated *like* for three of the twelve extreme items, which are marked as *like*, and rated *dislike* for two of the eight extreme items, which are marked as *dislike*, then it can be estimated that the ratings of this particular user is reversed. Figure 3.1 displays a sample reconstruction process with four extreme items at top. The values in these items are compared with corresponding items in each user's vectors if majority of corresponding ratings are in accordance with

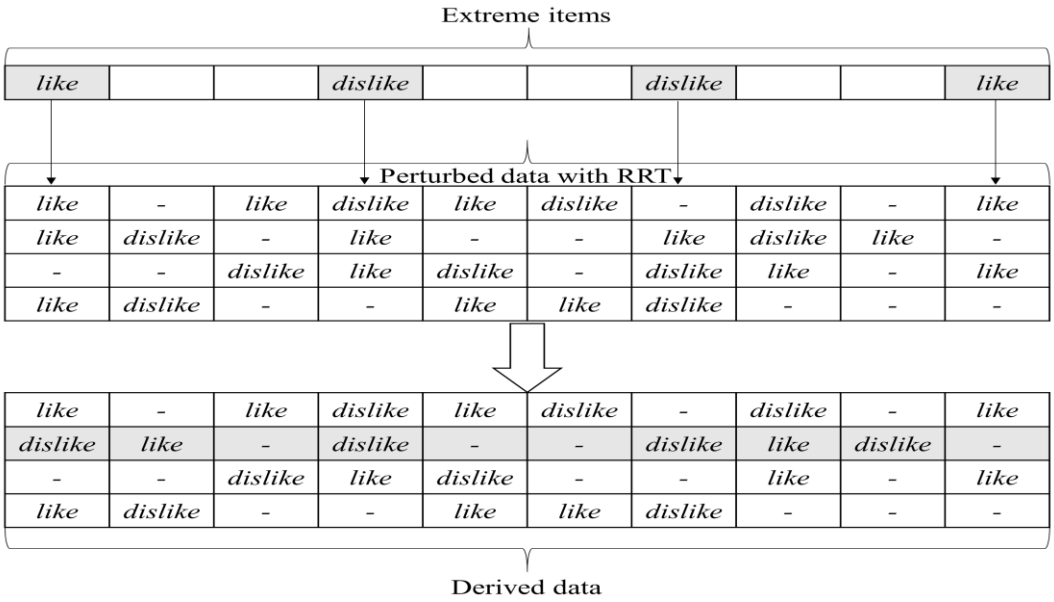


Figure 3.1. *Reconstruction with extreme items*

the extreme item ratings, then the user vector is preserved. Otherwise, it is reversed, which is highlighted in the derived data set.

The reconstruction method can be briefly described for the one-group scheme with constant θ as follows:

1. Apply Eq. 3.2 to calculate the popularity and unpopularity index values.
2. Sort items in descending order according to their popularity and unpopularity indexes.
3. Select the first *N* of the sorted items as extreme items.
4. Mark as *like* or *dislike* for those extreme items based on their popularity or unpopularity indexes, respectively.
5. For each user *u*, compare the ratings with extreme items.
    a. If the majority of the ratings are in accordance with the extreme items, then keep the rating vector as it is. This user preserves his or her rating vector.
    b. Otherwise, reverse the rating vector (transform *like*s into *dislike*s and *dislike*s into *like*s) to obtain the original vector.

3.1.1. Extending reconstruction model for multi-group

In addition to the one-group approach, users might use a multi-group scheme and can divide their rating vector into G groups as discussed in Chapter 2.1. Instead of

applying RRT for all items in the vector for once, users apply RRT G times for each group. Therefore, each group is, independently from each other, either preserved or reversed. As G grows, the privacy provided by the targeted PPCF scheme tightens (Polat and Du, 2006; Gambs and Lolive, 2013). Principally, the maximum privacy can be achieved when G is equal to *m*, the number of items, while the minimum privacy is the case where G is 1. With the maximum privacy, a disclosure about a rating only reveals the rating of the related item while a disclosure about a rating reveals whole rating vector of a user with the minimum privacy.

When there are G groups, the initial assumption about the calculation of items' popularity and unpopularity indexes (Eq. 3.2) to extract extreme items holds. The only difference is that extreme items need to be extracted for each group separately in a multi-group scheme. Thus, each group needs to be taken care of independently. First, extreme items are extracted for each group. After extreme items are determined, the ratings for all users are compared with extreme items in each group separately. If most of the votes are similar, items of that group are preserved. Otherwise, they are reversed.

Dealing each group independently might have some foreseeable problems for a small number of extreme items. If extreme items are extracted with no consideration of which groups they belong to, the majority of them can gather around a particular group. Suppose that fifty extreme items will be extracted for a 10-group scheme and thirty of them belong to the first group. Assume that the second group has the rest of twenty extreme items. The remaining eight groups would hold no extreme items. Such a case might dismiss the reconstruction of the groups with no extreme items. To avoid such an issue, equal number of extreme items can be extracted for each group. This approach might be advantageous for a relatively small number of extreme item sets and larger G values. When extreme items set is small, and G is large, each group must be guaranteed to have at least a fair number of extreme items to recover original ratings. On the other hand, if the extreme item set is large, the problem of gathering extreme items around a group might be less obvious because each group will probably have enough extreme items for data reconstruction. We will call the first approach, where the number of extreme items per group is not considered, as *classical approach* (CA). The second case, where extreme items will be shared almost equally between groups, will be called *fair approach* (FA) because it gives equal chances to each group to be reconstructed.

Another issue in RRT is to determine the value of θ. Notice that users might either use a fixed θ value or choose θ values uniformly randomly over the range (0.50, 1.0]. Hence, after presenting the extended method for the multi-group scheme, it should be discussed how to extend it if users select random θ values uniformly. The extension for random θ is given in the following subsection.

3.1.2. Extending reconstruction model for random θ

In case of random θ, using Eq. 3.2 becomes ambiguous to extract extreme items because there is no prior knowledge about θ and each user determines it randomly. Hence, an estimation of θ is needed. Determining θ values for each user is nothing more than a random guess because there is no prior knowledge about θ values except the fact that they are randomly generated using a uniform distribution from the range (0.5, 1.0]. Instead of trying to guess a unique θ for each user, an approach to determine a common θ value for all users based on the expected value of the uniform distribution is preferred.

When θ is not public, the expected value is a good candidate for an estimation about θ. The calculation of popularity and unpopularity indexes for an item will be performed column-wise throughout all users so expected value of θ can be utilized. This calculation is repeated for all items to extract the extreme items. Then, the same practice as explained in the algorithm proposed for the one-group scheme is followed to reconstruct the original data.

3.1.3. Exploiting significance weighting

An item is qualified to be an extreme item based on its rank in popularity and unpopularity indexes after Eq. 3.2. is applied. This extreme item extraction process does not take the number of users who rates an item into account. There might be some items that are rated by only a few users but end up being extracted as extreme items due to their high rank in popularity or unpopularity indexes. Such a case might promote some items with very few ratings but a relatively higher rank and ignore some other items that have many ratings but a relatively lower rank.

To promote popularity and unpopularity indexes, the idea of significance weighting, SW, can be employed. Herlocker and Konstan (1999) utilize SW to promote similarity values that are based on a large number of commonly rated items. They devalue a similarity value between users if it is calculated with less than 50 commonly rated items. A similar approach is utilized by Polat and Du (2006) as well. They set SW correlation

factor to 2*c* / *t* where *t* is equal to the total number of users, which is *n*, and *c* is the number of commonly ratings between items. If *c* is greater than *t* / 2, SW factor is set to 1. Otherwise, they multiply the similarity by SW correlation factor of 2*c* / *t*. In this dissertation, 2c / *t* will be used as SW, but *t* can be set any value rather than being fixed at *n*. *t* is a denominator to devalue popularity and unpopularity indexes based on the value of *c*. The SW factor is set to 1 *c* is greater than half of *t*. The slight difference between the SW correlation factor in this study and the one used by Polat and Du (2006) is that they apply SW on item-item similarities while it is applied on the popularity and unpopularity item indexes in extreme item extraction process.

Besides SW, a limit on the minimum number of users rating an item can be set. Items with more ratings than the limit are eligible for the extreme item extraction process. Items with fewer ratings than the limit are ignored. Therefore, items with enough ratings can be promoted while selecting extreme items. The basic idea behind these two approaches, SW and the minimum number of rating requirement, is to promote items with more ratings. These two approaches are analyzed in the experiments.

3.1.4. Exploiting auxiliary information

The fundamental reconstruction method is based on determining extreme items. It leverages such knowledge to derive actual binary ratings from a disguised data set. Although finding out extreme items is not a daunting task, some auxiliary information can be beneficial to the reconstruction approaches. Some publicly available auxiliary information can be a good candidate to become popular or unpopular items. Indeed, extreme item extraction aims to identify popular and unpopular items in the perturbed data set. Publicly available information can be integrated into the extraction process of the extreme items so that the proposed method is backed up with widely accepted data approved by many others. Inserting public and auxiliary information as additional elements shaped from the views of the great amount of people to improve the reconstruction approaches would be beneficial.

Since this dissertation focuses on deriving binary ratings in a movie related data set, public information about all movies included in the data set has been collected from a well-known movie website, IMDB[2], to make use of auxiliary information. The publicly available data is then attached to the reconstruction method. This method of integrating

publicly available information into the set of extreme items helps decide if a user has reversed or preserved his or her rating vector.

The method of integrating auxiliary public information into the extreme items is based on a simple idea of discovering potential items that the reconstruction algorithm is not aware of. Thus, public data items from IMDB that are not included in extreme items set are listed based on their average ratings. The items, which are retrieved from IMDB, whose average ratings greater than a predetermined popularity threshold are marked as *like* (high popularity index). Similarly, the items, which are retrieved from IMDB, with average ratings less than a predetermined unpopularity threshold are marked as *dislike* (high unpopularity index). These items, marked either *like* or *dislike*, are accepted as the new extreme items and merged with the original extreme items extracted from the perturbed data. Since there are three major applicable scenarios to implement the reconstruction method, it should be discussed how adding auxiliary data into the reconstruction method for such cases.

In the first case, one-group and constant θ, the integration of new extreme items obtained from auxiliary data is straightforward. Since there is no group to consider, extreme items are merged with the ones obtained from the auxiliary data, IMDB. After creating an extended extreme item set, the last step of the method is applied.

In the second case, multi-group scheme and constant θ, auxiliary information for each group must be handled. The items from IMDB in each separate group are ordered based on their public average ratings and potential items are marked as *like* or *dislike*, as explained previously. Then, items marked as *like* or *dislike* from each group are inserted into the extreme items set of the group to which they belong. Finally, the algorithm is applied to the extended extreme item set by taking the pattern of extreme items into account to obtain the estimated data set. The key point is to treat auxiliary items from each group separately.

The last case, where θ is randomly drawn from an interval, is more related to making a prediction about the expected value of θ. One must consider one- or multi-group cases while adapting the auxiliary information into this approach. If the one-group scheme is used with the random θ, the attacker must consider measures taken in the first case to exploit auxiliary information. Otherwise, the attacker should think about the second case, which is discussed in the previous paragraph. Note that the crucial point in exploiting auxiliary information is to cope with the number of groups instead of θ.

3.2. Reconstructing Rated Items

When discovering genuine rated items in the perturbed data set, exploiting noise elimination techniques can be useful if items are numerically rated for randomization process (Demirelli Okkalioglu et al., 2016). Randomization proposed by Polat and Du (2003) inserts some fake ratings into the numerically rated user-item matrix and this method of inserting fake ratings can be considered as noise to the original data. Noise elimination techniques can be useful to some extent to derive original ratings (Demirelli Okkalioglu, Koc and Polat, 2016; Gu, Wu and Li, 2006; 2008). On the other hand, in binary rated data, RRT possibly creates multiple groups and reverses ratings if randomly selected value by each user is greater than the predetermined θ value. RRT alters the characteristics of original data deeply. Additionally, fake ratings inserted into empty cells and their characteristic is not different from original ratings, which makes them difficult to discover. Inserted items are just *like*s and *dislike*s just like any other original items. Even RRT, which reverses some ratings, alone distorts the binary data pattern dramatically. In case of θ with 0.65, the perturbed data would have about 35 percent different ratings from the original matrix even though appended fake items are not considered. If the fake items are taken into account, the change between masked and original data becomes larger.

Although it is argued that noise elimination techniques do not help discover true rated items, exploiting public information could be useful. Since the primary focus is to identify genuinely rated items in the perturbed data, collecting auxiliary information could reveal a high degree of useful information. Auxiliary information is used about the targeted data set to test the hypothesis that auxiliary information would identify true rated items with decent accuracy. The data set is MLM with movie ratings of 3,883 users for 6,040 movies. Demographic data is already available with the data set.

3.2.1. Exploiting auxiliary information

The intuition to discover rated items is based on the idea of collecting public auxiliary information about the data set. While determining items that have been rated, the abovementioned auxiliary information will be used. Since there are fake items appended into user vectors, the first step of the algorithm must figure out how many items in the perturbed data are indeed rated in the original data. The server might want to find out the number of ratings made by each user; however, the filling method is based on user

vector density, d, as discussed in Chapter 2. Such an estimation can be achieved by a random guess due to random nature of the filling method. Instead of trying to estimate the number of rated items for each user, a vertical view to estimate the number of genuine users who rate the related item would be more practical. For such an attempt, the first step must be identifying the density of the data set, d_{set}, before the filling method is applied. d_{set} can be easily estimated because appended ratings will be approximately the half of the genuine ratings which originally reside in the original data matrix of users. Because the server has the perturbed data set with all genuine and appended fake ratings, it is easy to figure out for the server that d_{set} is approximately 66% of the density of the perturbed data set.

After estimating d_{set}, the server must determine the number of users who rated the related item. It is obvious that the total number of users in the perturbed data matrix whose i-th item is rated, r_i, is composed of the number of genuine, r_{gi}, and fake, r_{fi}, raters of the related item, $r_i = r_{gi} + r_{fi}$. The server needs to calculate r_{gi}; however, r_{fi} is an unknown. r_{fi} can be estimated by using the expected value of the filling factor. The filling factor is associated with the d_{set} because ratings are appended based on d, which is the user density. Although identifying the filling factor for each user requires a random guess as discussed in the previous paragraph, it can be estimated for each item. When each item is examined independently, the filling factor will approach to d_{set} / 2 because all users apply the filling method in terms of their density, d. As a result, r_{fi} can be estimated in terms of r_{gi}, d_{set} and n, the number of users, as in Eq. 3.3. After estimating r_{fi}, the estimation of r_{gi} is trivial.

$$\begin{aligned} r_{fi} &\approx \left(n - r_{gi}\right) \times \frac{d_{set}}{2} \\ r_{gi} &\approx \frac{2 \times r_i - n \times d_{set}}{2 - d_{set}} \end{aligned} \tag{3.3}$$

The algorithm must identify genuine items rated by users after estimating r_{gi}. At this point, auxiliary public information might be helpful to discover these items. Since users will be marked as genuine or fake raters of each item in the perturbed data set, user-related auxiliary information about movies would be more helpful. The related data set used in this dissertation contains demographic information about its users. Therefore, the demographic user data is integrated along with movie genres data believing that people's taste of movie-genre differ by age group. The basic intuition is that some age groups are more willing to watch some movie genres and a report conducted for British Film Institute

analyzing contributions of movies to United Kingdom culture is utilized[3]. The algorithm to discover the rated items has three-phases.

1. First, genres of each movie are listed (comedy, action, adventure, drama, etc.). A movie might have multiple genres. This information comes with the data set.
2. Then, age groups of users are determined, and a user cell remains rated if age group and movie genre relation holds. For example, if a user is young, it is assumed that he or she likes comedy. There are a couple of rules for each age group inspired by a report conducted for British Film Institute[3].
3. After this process, there still might be some unidentified user cells. Remember that r_{gi} is determined. The number of ratings made by each user in the perturbed matrix is listed to locate the remaining users who might have rated the movie. A higher number of ratings for a user in the perturbed matrix can be an indication of a specific user who has also rated for the related movie. This step of the algorithm to mark the remaining users as rated is based on the number of ratings they have in the disguised data.

3.3. Experiments

Various experiments have been conducted to test how varying of different parameters in the central server-based PPCF schemes affect the reconstruction results. Experiments are grouped in two different subtitles based on two aspects of privacy. Unless otherwise stated, the θ is constant and set to 0.65 while G is 5. The number of extreme items will be experimentally determined in the first experiment, and the selected number of extreme items will be utilized afterward.

3.3.1. Reconstructing actual item ratings

This group of experiments aims to reconstruct rating values from the perturbed data, which deals with the first aspect of privacy. There are seven experiments in this section. Experiments start with controlling how the number of extreme items has an effect on the reconstruction approaches. Since θ is a variable that determines whether a user should

[3] This report by Northern Allience and Ipsos MediaCT is available at http://old.bfi.org.uk/publications/openingoureyes/downloads/Appendix-2-Results-Tables-Cultural-Contribution-of-Film.pdf

keep or reserve his or her ratings, the server approximately has 100 × θ percent of the original data. Therefore, any reconstruction result should be evaluated considering this granted ratio. Privacy measures, θ and G, proposed in targeted studies (Polat and Du, 2006; Kaleli and Polat, 2007b) are controlled in following three experiments. These measures are examined to analyze the argument that tighter privacy measures help protect confidential data. The fifth experiment discusses the effects of the number of users participating in a PPCF process in terms of reconstruction output. Then, SW and the number of minimum ratings to be eligible for extreme item nomination is analyzed. This study argues that exploiting auxiliary information could help recover the original data and utilizes movie-related information collected from IMDB. In the last experiment, the effects of introducing such auxiliary information are discussed. Throughout the experiment in this subsection, only *prec* values will be given. Since there is no filling method included for this part, it is not possible that any item rated either *like* or *dislike* will be marked as unrated by users. Table 1.2 gives the confusion matrix, and it is clear that applying RRT without a filling method would result in the same result for *prec* and *rec*.

3.3.1.1. *Effects of varying number of extreme items*

The first experiment is conducted to analyze the effects of the number of extreme items on the reconstruction. There are two reconstruction approaches, FA and CA. The main idea in CA is to extract the best extreme items and apply the reconstruction. CA does not consider the distribution of these extreme items among groups. On the other hand, in FA, the initial argument is that extreme items should be split equally between groups, which is neglected in CA, so that each group is given an opportunity for the reconstruction. The small number of extreme items might lack enough data for the reconstruction; the initial hypothesis is that increasing number of extreme items will provide better *prec* results. In terms of FA and CA, FA might be better than CA because FA tries to reconstruct each group by letting them have an equal number of extreme items. The number of extreme items in this experiment is associated with *m*, so it is varied between the factors of *m*, *m*/32 (121), *m*/16 (242), *m*/8 (485), *m*/4 (970), *m*/2 (1,941), 3*m*/4 (2,912), and 7*m*/8 (3,397). Overall averages of *prec* values are displayed in Figure 3.2. The corresponding values for the number of extreme items are given in x-axis of the figure.

As seen in Figure 3.2, FA and CA start with very close *prec* values to each other and 0.650, which is the expected *prec* ratio due to RRT. This could be an indication that the reconstruction approaches need more extreme items to derive meaningful data. The trend of increase in FA and CA with the larger number of extreme items supports this fact. FA demonstrates a clear increasing trend until 1,941 extreme items. FA reaches its peak *prec* value at 1,941with 0.948. After this point, FA shows a stable trend with insignificant declines in 2,912 and 3,397 extreme items. The behavior of FA with varying number of extreme items shows that FA achieves a good accuracy in terms of *prec* for larger extreme items. In terms of CA, there is a consistent and sharp increase until 970 extreme items are exploited. The highest *prec* recorded for CA is 0.945 when 970 extreme items are utilized. Larger extreme item sets beyond this point remain relatively stable with slight decreases as the number of extreme items grows.

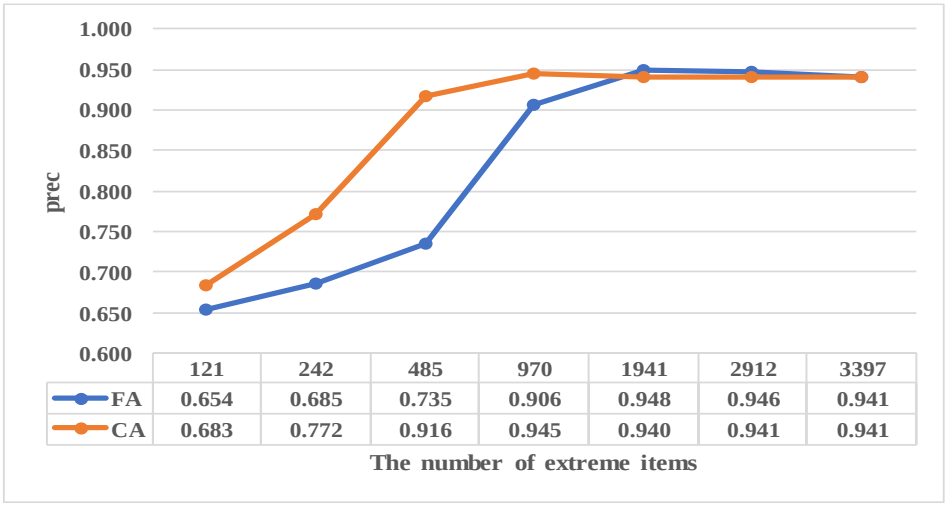


Figure 3.2. *Reconstruction with varying number of extreme items*

When FA is compared with CA, it is clear that CA records much better results until 970 extreme items are used in the reconstruction. FA slightly surpasses CA starting with 1,941 through 3,397 extreme items. Contrary to the initial expectation, FA could not be considered useful compared with CA until 1,941 extreme items. This experiment shows that increasing number of extreme items provides sharp increases in terms of *prec*. However, this trend is stabilized after a certain point for both of the approaches. The reason behind this phenomenon could be attributed to the fact that the reconstruction approaches could not find out undiscovered and useful extreme items after a certain point, which seems to be 1,941 and 970 for FA and CA, respectively.

Because 1,941 and 970 extreme items mark the best output for FA and CA, respectively, these numbers will be used in the following experiments as the default value for the number of extreme items.

3.3.1.2. *Effects of varying θ*

The second experiment is conducted on varying θ values. θ determines a threshold to preserve or reverse ratings in a group based on a uniformly drawn random number. Therefore, RRT approximately assures the server that $100 \times \theta$ percent of the perturbed data is indeed original, but it does not tell which ratings are preserved. This means that without any reconstruction effort, *prec* would yield roughly 0.650. In the targeted study, Polat and Du (2006) vary θ between 0.51 and 1.00. As an expectation for this experiment, as θ goes away from 0.51 to 1.00, randomness starts to vanish, so FA and CA would produce higher *prec* results. Figure 3.3 displays the results. The dashed line demonstrates the expected precision value of the perturbed data due to RRT, which is associated with θ. It is a linear line starting from 0.51 up to 0.95.

The first thing that draws attention when Figure 3.3 is analyzed is that the difference in terms of *prec* between the expected precision and the reconstruction approaches are very wide except θ is 0.95, which is a very impractical value to be set as a privacy measure. When θ is 0.51, where the perturbed data is at its highest randomized version with 0.510 expected precision, FA and CA perform *prec* results as high as 0.674 and 0.602, respectively. A marked increase is recorded for FA and CA when θ is 0.55. At this point, FA and CA record 0.876 and 0.884 as *prec* values, respectively, which is the highest difference recorded between the reconstruction approaches and expected precision. Beyond this point, the difference in *prec* values compared to the expected precision is still dramatic until θ is 0.75, which is at least roughly as much as 0.200. As randomness greatly diminishes from 0.85 to 0.95, where RRT's data perturbation effect weakens as well, the difference in *prec* between the reconstruction approaches and expected precision starts to narrow. When θ is 0.95, FA and CA achieve 0.961 and 0.955 in terms of *prec*, respectively, which are still marginally greater than the expected precision.

θ values closer to 0.51 mean the perturbed data has more randomness due to RRT, as repeatedly discussed. The reconstruction results are improving as θ moves away from 0.51; however, this also means that randomness diminishes. Therefore, the evaluation

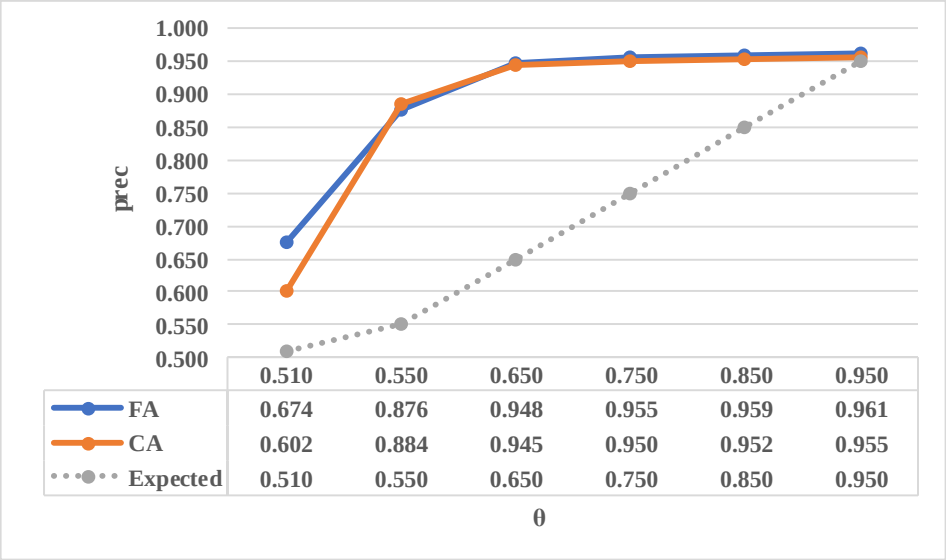


Figure 3.3. *Reconstruction with varying θ*

criterion, *prec*, is compared with the expected precision. In all cases, proposed reconstruction approaches always beat the expected precision.

3.3.1.3. *Effects of random θ*

Another factor that might affect the reconstruction is the way of selecting θ values. Recall that users can use a constant θ value, as considered in the previous experiments. Each user can also uniformly randomly choose θ values from the range (0.5, 1.0]. This experiment is conducted to evaluate the effects of selecting θ values randomly by each user independently. In this experiment, it is assumed that each user can randomly select a uniform θ value over the range (0.50, 1.0] rather than using a constant one. As discussed in subsection Chapter 3.1.2, an expected value for θ needs to be set, and it is set to 0.755. The results of this experiment are compared with a base case, where θ is constant and set to 0.755. Overall averages are displayed in Table. 3.1.

Table 3.1. *Comparison of reconstruction with random and constant θ*

Recons. Approach	Random θ	Constant θ = 0.755
FA	0.941	0.956
CA	0.916	0.951

As can be seen in Table. 3.1, using a constant θ value yields modestly higher results compared to the case, where random θ values are uniformly selected. This phenomenon is expected because it is always guaranteed to use the true value of θ in Eq. 3.2 when users utilize a constant θ value. However, θ can only be estimated with the random case.

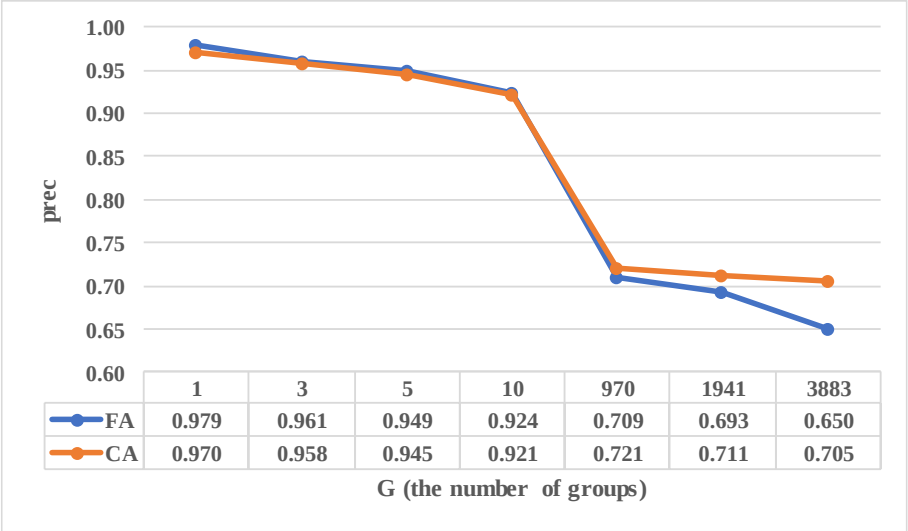
Although, constant θ case beats the random θ case, as expected, utilizing the expected value of θ produces promising results as well. *prec* values recorded with random θ case are as high as 0.941 and 0.916 for FA and CA, respectively. These results are considerably higher than the expected reconstruction precision which could be calculated approximately as $100 \times \tilde{\theta}$, where $\tilde{\theta}$ is the expected value for random θ for whole data set, and this calculation yields 0.755. This is a prominent outcome because there is no prior knowledge about θ except its possible range of (0.50, 1.0].

3.3.1.4. *Effects of varying G*

This experiment scrutinizes how varying G affects the reconstruction approaches. As G grows, users split their data more, and each group is independently subject to RRT. It is obvious that larger G adds randomness to the original data; intuitively, as randomness increases, the reconstruction criterion should decrease. Therefore, the hypothesis is that *prec* values will perform a decline as G gets larger. The reason why larger G values provide more privacy for users is that the number of items belonging to each group is small (Gambs and Lolive, 2013). In case of disclosure of a rating in a group causes a comprise in the privacy of items that are in the same group with the compromised item. G is varied among 1, 3, 5, 10, 970 (m/4), 1,941 (m/2), 3,883 (m) to demonstrate the performance of the reconstruction approaches between the maximum and minimum privacy. Figure 3.4 illustrates the results.

FA and CA record very close *prec* results and are performing a stable decline as G grows. Although the decline is important, each case except FA when G is 3,883 surpasses the expected *prec* results. Recall that θ is 0.65 and the expected *prec* is 0.650. Especially, *prec* is very promising for the one-group scheme, which is slightly above 0.979 and 0.970 for FA and CA, respectively. The decline is very sharp when G reaches 970 (m/4). G being 970 means that each group has only four items; therefore, qualified extreme items will only help four items be reconstructed. Since the number of items that each group shrinks with larger G, a downward trend continues toward 3,883-group scheme.

To sum up, this experiment shows G is an important factor to offer privacy for individuals. Nonetheless, as privacy increases, accuracy diminishes. This phrase is a leitmotif describing the fundamental tradeoff of PPCF. Therefore, very large G values are not very practical for CF systems. The reconstruction approaches; on the other hand, achieve very promising results up to 10-group, which are practical numbers of groups due



	1	3	5	10	970	1941	3883
FA	0.979	0.961	0.949	0.924	0.709	0.693	0.650
CA	0.970	0.958	0.945	0.921	0.721	0.711	0.705

Figure 3.4. *Reconstruction with varying G*

to accuracy concerns in PPCF. Polat and Du (2006) and Kaleli and Polat (2007b) test their PPCF schemes up to 5-group; therefore, 10 could be considered a very tight privacy measure and *prec* results are above 0.920 for FA and CA for 10-groups. When this result is compared with the expected *prec*, which is indeed 0.650 in this experiment, there is a recovery more than 0.270 for FA and CA.

3.3.1.5. *Effects of varying n*

n, the number of users, participated in the reconstruction process might be another factor that would affect the reconstruction. Intuitively, increasing *n* should contribute to a higher *prec*. When *n* increases, the perturbed data matrix would contain more ratings for items. The more ratings the reconstruction approaches work with, the higher the possibility of extracting qualified extreme items will be. To test the hypothesis, an experiment with varying *n* from *n*/32 (188), *n*/16 (377), *n*/8 (755), *n*/4 (1,510), *n*/2 (3,020), and *n* (6,040) is carried out. The averages of *prec* values are given in Figure 3.5.

Empirical outcomes in Figure 3.5 show that *prec* improves with increasing *n* values. Such outcomes verify the hypothesis, and they support the intuition that more contribution of users produces qualified extreme items. A consistent upward trend in *prec* through increasing *n* values is clear. Although improvements follow a relatively stable trend for *n* values larger than 1,510, enhancement in *prec* is dramatically significant compared with the smaller *n* values. At this point, *prec* for FA and CA is recorded as 0.748 and 0.749 when *n* is 188, respectively. The two approaches perform *prec* value of

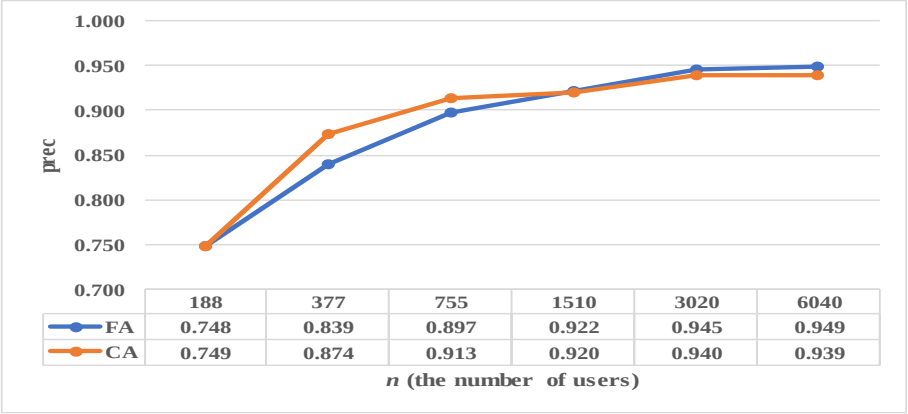


Figure 3.5. *Reconstruction with varying n*

0.949 and 0.939 for *n* is 6040. Such a difference in recovery between *n* being 188 and 6,040 marks an enhancement of 0.201 and 0.190 in *prec* for FA and CA, respectively. In summary, overall trend illustrates that increasing *n* helps *prec*.

3.3.1.6. *Effects of SW and the limit on the minimum number of ratings in extreme item extraction with varying number of extreme items*

SW and the limit on the minimum number of ratings have similar foundations; therefore, they are grouped in here and tested together. This experiment discusses the effects of both approaches in reconstruction. The hypothesis is that these approaches might be beneficial for cases where extreme item set is small. Extreme item set starts to include many items as it gets larger; therefore, the effect of these approaches is expected to wear off. Apart from the size of extreme item sets, another important point would be the limit set on the number of ratings. Since traditional CF schemes are sparse, an important deal of items will be eliminated from the extreme item extraction process if the minimum number of ratings set for extreme item eligibility is a large number. Increasing the minimum number of ratings is expected to have a diminishing effect on results since the number of items satisfying this criterion will dramatically drop. This experiment is performed by varying number extreme items to see the effects of these approaches with small and large extreme item sets. Note that SW correlation factor is $2c$ / t. t and the minimum number of ratings is associated with n and varied between 0, n/32 (188), n/16 (377), n/8 (755), n/4 (1,510), and n/2 (3,020). t is also set to n (6,040). However, the minimum number of ratings is not set to n because no item can be eligible to be an extreme item in such a case. The column set to 0 means that the base method of extraction of extreme items is applied. Results are listed in Table 3.2 and 3.3.

Remember that the optimum number of extreme items are 1,941 and 970 for FA and CA, respectively, as discussed in the first experiment in this part. The results show that SW correlation factor and the limit on the minimum number of ratings records an improvement in *prec* up to the optimum number of extreme items. This means that promoting items with a relatively high number of ratings could help in reconstruction unless extreme item set is large while discovering extreme items. The reason why the positive effect of the approaches with a large extreme item is because such large set of extreme items has enough variety of items to reconstruct the original data.

Table 3.2. *Reconstruction with varying limit on the minimum number of ratings*

Recons. Appr.	Number of extreme items	The minimum number of ratings					
		0	**188**	**377**	**775**	**1540**	**3020**
FA	121	0.656	0.878	**0.910**	0.894	0.819	0.583
	242	0.679	0.925	**0.927**	0.884	0.779	0.573
	485	0.740	**0.940**	0.928	0.844	0.739	0.564
	970	0.904	**0.938**	0.906	0.798	0.707	0.614
	1941	**0.948**	0.923	0.857	0.744	0.672	0.631
	2912	**0.946**	0.901	0.805	0.703	0.662	0.652
	3397	**0.943**	0.891	0.788	0.691	0.655	0.652
CA	121	0.685	**0.919**	0.908	0.830	0.836	0.678
	242	0.770	**0.939**	0.914	0.889	0.818	0.664
	485	0.917	**0.938**	0.918	0.857	0.801	0.664
	970	**0.946**	0.931	0.909	0.826	0.788	0.702
	1941	**0.939**	0.919	0.865	0.779	0.730	0.703
	2912	**0.941**	0.900	0.810	0.706	0.666	0.655
	3397	**0.940**	0.891	0.789	0.692	0.655	0.651

Notice in Table 3.2 that large *t* values have relatively better effect with small extreme item set. To illustrate, when the number of the extreme items is 121 and 242, the best scoring *t* value in terms of *prec* is 1540 while *t* is 755, 377 and 188 for 485, 970 and 1941 extreme items for FA. On the other hand, if the limit on the minimum number of ratings is a large number; the improvement starts to vanish as well. This occurs due to the fact that locating an item with so many ratings is so rare. For example, an item with more than 3,020 ratings means that half of the users have a rating for that item, which is not very likely in CF systems. To sum up, methods to promote items with more ratings while extracting extreme items is beneficial if the extreme item set is small. However, it does not help when the optimum number of extreme items are utilized which are 1,941 and 970 for FA and CA, respectively.

Table 3.3. *Reconstruction with the number set as denominator, t*

Recons. Appr.	Number of extr. items	*t*, the number set as denominator						
		0	**188**	**377**	**775**	**1540**	**3020**	**6040**
FA	121	0.656	0.723	0.830	0.880	**0.910**	0.901	0.879
	242	0.679	0.794	0.888	0.924	**0.928**	0.905	0.881
	485	0.740	0.886	0.935	**0.941**	0.935	0.906	0.907
	970	0.904	0.941	**0.948**	0.945	0.929	0.924	0.926
	1941	0.948	**0.949**	0.943	0.937	0.938	0.937	0.937
	2912	**0.946**	0.942	0.940	0.939	0.940	0.941	0.939
	3397	**0.943**	0.940	0.940	0.939	0.940	0.941	0.941
CA	121	0.685	0.816	0.896	0.912	**0.920**	0.898	0.878
	242	0.770	0.908	0.935	**0.939**	0.931	0.905	0.911
	485	0.917	0.941	**0.948**	0.946	0.932	0.928	0.923
	970	0.946	**0.947**	0.944	0.938	0.936	0.933	0.933
	1941	0.939	**0.940**	0.939	0.939	0.938	0.938	0.938
	2912	**0.941**	0.939	**0.941**	0.940	0.940	0.940	0.940
	3397	**0.940**	**0.940**	**0.940**	**0.940**	0.939	0.939	0.939

3.3.1.7. *Effects of auxiliary information with varying number of extreme items*

This experiment analyzes how integrating publicly available auxiliary information into the base reconstruction methods affects the overall performance. The hypothesis is that auxiliary information can contribute toward achieving better *prec* values. There might be some items among 3,883 available items that the base method could not manage to include into its extreme items set. Nonetheless, recall that number of extreme items is picked as 1,941 and 940 for FA and CA, respectively, in the base method in the first experiment analyzing the optimum number of extreme items. Integrating the auxiliary information, which is generally public, could be helpful to choose some undiscovered extreme items. 1,941 and 940 are already large numbers to include enough data for the reconstruction. However, a small number of extreme items could miss some useful items for the reconstruction. This experiment will test if including auxiliary item could help the reconstruction by utilizing a varying number of extreme items. Now, the hypothesis is that auxiliary information can contribute toward achieving better *prec* values, especially for small number extreme items because they might neglect some useful items while determining extreme items.

Movie related public information is retrieved from IMDB. Movies with an average rating greater than or equal to 8 as popular movies. Similarly, movies with an average rating less than 5 are accepted as unpopular movies. This public information is inserted into the extreme items set that are extracted from the perturbed data set. The experiment contains five cases related to auxiliary and public information and one case with the base

method. The first case is a test case for a base method with no auxiliary information. In the second case, only popular movies are used as auxiliary information. In the third case, unpopular movies are utilized while the fourth case contains Oscar winner movies. The fifth case utilizes a combination of the second and third cases while the last case utilizes the combination of the second, third and fourth cases. The outcomes are given in Table. 3.4 and the type of auxiliary information utilized given in the column headers of the table. Additionally, the largest *prec* value is marked bold for each row.

Empirical outcomes in Table 3.4 demonstrate that using publicly available auxiliary information can make a noteworthy contribution based on the number of extreme items utilized. The contribution of auxiliary information is very dominant for FA and CA if a small number of extreme items is utilized. Auxiliary information takes a critical role in recovering original data when the number of extreme items is up to 485 for both approaches. However, when the extreme item set is larger, the auxiliary information makes almost no help. Its contribution is very marginal to notice. The reason behind this phenomenon is that small number of extreme item set might have some missing items that could be important in terms of deriving the original data. In other words, some items might not be extracted as extreme items by the reconstruction approaches, but they might have an important effect that remains undiscovered. Auxiliary information with a small number of extreme items has a promising effect approaching the best cases with the base method. Extracting a high number of extreme items adds computational costs for very large data sets; hence, this experiment presents that utilizing auxiliary with relatively

Table 3.4. *Reconstruction with auxiliary information*

Recons. Appr.	Number of Extreme Items	Base Method	Popular	Unpopular	Oscar Won	Popular & Unpopular	All
FA	121	0.655	0.889	0.811	0.829	**0.925**	0.907
	242	0.687	0.893	0.815	0.835	**0.926**	0.908
	485	0.738	0.905	0.814	0.855	**0.928**	0.911
	970	0.904	0.929	0.914	0.908	**0.936**	0.922
	1941	0.948	0.948	**0.949**	0.940	**0.949**	0.941
	2912	0.946	0.946	**0.947**	0.942	0.946	0.942
	3397	**0.943**	0.941	0.942	0.939	0.942	0.939
CA	121	0.690	0.893	0.808	0.835	**0.926**	0.908
	242	0.772	0.901	0.849	0.850	**0.927**	0.910
	485	0.912	0.924	0.928	0.896	**0.934**	0.919
	970	0.945	0.944	0.946	0.935	**0.946**	0.937
	1941	0.940	0.940	**0.941**	0.937	0.940	0.938
	2912	0.940	0.940	**0.941**	0.938	0.940	0.937
	3397	0.940	0.940	**0.941**	0.937	**0.941**	0.938

small number of extreme items could be useful instead of exploiting a large number of extreme items for accurate reconstruction.

3.3.2. Reconstructing rated items

This experiment is conducted to derive if a rated item in the perturbed data set is really rated in the original data set or a fake item inserted by the filling method, which is the second aspect of privacy. Auxiliary information is utilized to create rules to accomplish this task. In the targeted schemes (Polat and Du, 2006; Kaleli and Polat, 2007b), users insert fake ratings by drawing a random number between (0, *d*], where *d* is the density of the vector. This filling method assures that a user can fill his or her rating vector with items less than the number of original ratings. The idea here is to find out the genuine ratings made by users. Since fake ratings are inserted into the original user vector based on a random process, the auxiliary information is utilized to derive genuine rated item. The report by British Film Institute[3] includes movie genre preference of people by different age groups. The related data set used in this dissertation is MLM, and it also contains demographic user information about user age groups. The results of the survey are matched with the age groups in MLM, and related rules are created to determine the genuine rated items. Contrary to the approaches deriving actual ratings, *prec* and *rec* would not yield same results in this experiment. Both are given for this experiment in Table 3.5. This experiment deal with the second aspect of privacy and evaluation criteria are calculated by the related formula given in Eq. 1.1.

The parameters, θ and G, has no effect on discovering rated items because this reconstruction is concerned about the filling method. The filling method executes its operation by inserting fake ratings. *prec* can be considered more important than *rec* in this experiment. Since the removal of items is not considered by the filling method, *rec* is 1.000 after the filling method is applied. Recall that this protocol hides the original ratings by appending new ones. Moreover, the reconstruction method achieves 0.786 in terms of *prec*. In the perturbed data, the *prec* could be estimated around 0.666 because every one item out of three in the perturbed data is fake due to the filling method. Recall that each user fills his or her rating vector with a random percentage drawn uniformly

Table 3.5. *Reconstruction of rated items*

prec	rec
0.786	0.804

from the range (0, d]. Thus, the density of fake items approximately becomes d_{set} / 2 over the whole data set that causes the estimated *prec* value to become 0.666. Indeed, this estimation of *prec* is a paraphrase of the calculation of the estimated density of the original data, d_{set}, in Chapter 3.2.1.

3.3.3. Reconstructing actual ratings from full-privacy

The previous two group of experiments handle the recovery of the original ratings from perturbed data in two different cases by applying data masking and hiding methods independently. In Chapter 3.3.1, data hiding method is not considered in the experiments. The purpose was to reconstruct data does not consider the second aspect of privacy. Likewise, the reconstruction in the experiment of Chapter 3.3.2 does not consider the first aspect of privacy; in other words, RRT with multi-group. In this experiment, both RRT with multi-group, which is the first aspect of privacy, and data hiding method, which is the second aspect of privacy, are applied together. In such a case, a malicious data holder needs to find out genuinely rated items before determining the actual rating values. Therefore, auxiliary information will first be utilized, and then extreme items will be extracted to derive the rating values of the original data. θ is set 0.650, and G is set to 5. The number of extreme items are 1,941 and 970 for FA and CA, respectively, without considering SW, the number of minimum ratings in extreme item extraction and auxiliary information.

Table 3.6 displays the results. The column "Granted" shows the *prec* and *rec* values calculated from the perturbed data with full-privacy before the reconstruction has been performed for comparison purposes. The reconstruction of the original data achieves higher results when compared with the granted ratio of both metrics. A malicious server has granted ratio of 0.431 *prec* and 0.650 *rec*. After the reconstruction is applied, *prec* is 0.739 and 0.738 for FA and CA, respectively. The difference between granted *prec* and the *prec* from FA and CA is significant and 0.308 and 0.307, respectively. *Rec* is very close for FA and CA and records 0.759, 0.758, respectively. These values are again remarkable when the granted *rec* value is considered. Note that it is impossible to exceed *prec* and *rec* values in Table 3.5 because deriving which items are rated is the first step

Table 3.6. *Reconstruction from full-privacy*

	FA	CA	Granted
prec	0.739	0.738	0.431
rec	0.759	0.758	0.650

of a two-step process when deriving original rating values from full-privacy. The final reconstruction results in Table 3.6 (with full-privacy) are very close to the results in Table 3.5. As a result, even if users perturb their data with full privacy, proposed reconstruction methods achieve remarkable results.

3.4. Conclusion

In this chapter, central server-based binary PPCF schemes (Polat and Du, 2006; Kaleli and Polat, 2007b) are examined. These schemes offer privacy preservation for their users by employing RRT with multi-group to disguise items' rating values and a filling method to hide the items rated by users. The attacks proposed in this chapter derive confidential user data by trying to reconstruct the original data matrix from the perturbed data that users submit to the server. The first attack extracts extreme items whose ratings are overwhelmingly consistent among different user by employing RRT estimation. Then, actual rating values are checked against these extreme items. The second attack utilizes auxiliary public information to derive the presence of a rating. The third attack derives original ratings when both data masking (RRT with multi-group) and data hiding are applied by integrating solutions from first two attacks.

Experiments about deriving the actual ratings analyze the targeted schemes with different control parameters. In the first experiment, the effects of the number of extreme items are analyzed. The results in this experiment are parallel with the hypothesis that the reconstruction approaches need enough data for data recovery. Therefore, as the number of extreme items increase, higher reconstruction results are achieved up to a certain point. Then, reconstruction results follow a rather stable trend with marginal declines. In the following experiment, reconstruction approaches are tested against varying θ value. θ is used as a determiner to preserve or reverse ratings. The randomness introduced by θ reaches its highest ratio as it approaches 0.500. Although the experimental results were consistent with the fact that reconstruction results decrease as randomness increases, proposed reconstruction approaches provide very high reconstruction results especially when $\theta = 0.650$, which is a practical value for a prediction process. Polat and Du (2006) shows in their work that accuracy diminishes as θ values approach 0.50. As discussed before, privacy and accuracy need a balance to avoid dramatic loss in accuracy. The next experiment is tested how setting θ randomly for each user affects the reconstruction. In this setting, the server is unaware of the exact θ value used by users except for its range.

The proposed solution is to use the expected value of θ and the outcoming results are compared with the case where θ is constant. For this comparison, θ is set to the expected value while employing random θ. As expected, constant θ is better than random θ case but using expected value of θ yield decent results. Apart from θ, G is another factor that complements RRT approach. G is varied between 1 and m, the number of items, and the experimental results are in line with the expectation that 1-group scheme is the most prone to privacy disclosure while the m-group scheme is the most resilient in terms of privacy disclosure. The number of users, n, is also tested to see how varying values of it has an effect on the reconstruction. It is discussed that more users are needed for the better nomination of extreme items, and increasing values of n have a contribution. Then, SW and the minimum number of ratings to be eligible for extreme item nomination is tested. These parameters generally help for small extreme item sets. The last experiment about deriving actual ratings is to control whether integrating auxiliary information into the base method can be a factor promoting the reconstruction. The experiment shows that it can be a prominent factor if the number of extreme items are small and yields very close precision results with cases where a high number of extreme items are utilized. However, the effect of auxiliary information integration vanishes as the extreme item set gets larger. Such a help from auxiliary information can be exploited for big data sets where extracting a high number of extreme items could be costly. Recall that FA and CA require $m/2$ and $m/4$ number of extreme items for MLM data set, respectively, for the best reconstruction results.

The motivation for the second attack is to determine if a rated item in the perturbed data set received by the server is a genuine or fake one inserted by the filling method. Since this operation is merely based on a random appending of fake items, the auxiliary information is exploited for reconstruction. The results show that exploiting auxiliary information could derive genuine rated ratings from the perturbed data that is disguised by the filling method.

The last attack aims to derive private individual rating values when data hiding method is integrated. The attack technique first applies the auxiliary information to derive which items are rated. Then, extreme item extraction process recovers original rating values. The results demonstrate remarkable results when compared to the granted metrics of the perturbed data.

To summarize, RRT with multi-group is prone to reconstruction with decent accuracy unless full-privacy is applied. Even full-privacy is applied, the reconstruction of actual rating values, the first aspect of privacy, is remarkable. The problem with RRT in central server-based PPCF is that it is originally designed to reveal percentages of binary preferences. Such a disclosure is an extreme vulnerability that can be exploited.

4. DERIVING PRIVATE DATA FROM BINARY DISTRIBUTED PPCF SCHEMES

The previous chapter deals with the central server-based binary PPCF scheme (Polat and Du, 2006), where users perturb their data before sending a central server. This chapter handles the case where user data does not reside on a central server. Instead, multiple parties hold different parts of user data. A data holder can collaborate with others to enhance its own data set. Such a collaboration could happen between two- or multi-parties either horizontally or vertically as discussed in Chapter 1 and 2. Unlike central server-based PPCF schemes, users send their original data to a data holder, and the data holder perturbs the collected original user data to preserve the privacy of institutional data from any malicious party. The effort of protecting the privacy of institutional data is called institutional privacy and discussed in Chapter 1.

The objective of this chapter is to derive the private institutional data from the binary P3CF (Polat and Du, 2005c; 2008; Kaleli and Polat, 2007a) or PPDCF (Polat and Du, 2008; Kaleli and Polat, 2015) schemes in the presence of a malicious party. The related PPCF schemes in this chapter are given as preliminaries in Chapter 2. Although P3CF schemes are built upon in the presence of a two-party in the PPCF community, they are included in this chapter together with PPDCF schemes. This way of categorization is preferred because many PPDCF schemes discussed in this chapter are extended from P3CF schemes and the attacks deriving private data can be applied on both P3CF and PPDCF schemes together. The following section introduces attack techniques and their applicability on PPDCF schemes.

4.1. Attacks to Derive Private Data

Attack techniques in this chapter aim to reconstruct original data matrices of parties taking part in various distributed binary PPCF schemes. There are four attack techniques with different motivations. To summarize, the first one picks an item and alienates it from other items' rating in the query and monitors the similarities returned from other parties for the query. The second attack monitors perfect matches with the query. While the third one manipulates a different item's rating, each time a new query is sent. The last attack exploits neighborhood relation when the history of a user is known. The attacks in this chapter are designed as if no privacy measures were taken. In other words, these attacks

will be covered here for distributed binary CF protocols. Then, privacy measures are applied to show how these parameters help preserve privacy.

4.1.1. Alienate the victim attack

Alienate the victim item attack is designed to derive confidential data from NBC-based distributed binary PPCF schemes. The basic principle in *alienate the victim* attack is to mark a victim item with a different binary rating from the rest. A malicious party prepares an active query in a way that an item is alienated with its rating from the remaining rated items. The objective is to derive the original rating of the victim item. Chapter 2 introduces the binary PPCF schemes targeted in this dissertation. In the NBC-based binary PPCF schemes (Kaleli and Polat, 2015), parties are not able to transfer similarity values in the aggregated form. Recall that parties send $p(q \mid like)$ and $p(q \mid dislike)$ results for each user they own to the master party for the final similarity calculation. Furthermore, the numerator, $PN_{ig,}$ and the denominator parts, PD_{ig}, are separately transferred to the master party due to the fact that a query might have multi-groups. Such an exchange between parties can be exploited because collaborating parties calculate the related similarity value for the victim item by comparing the victim item's rating with the corresponding rating in users' vector. Therefore, the master party can infer the rating for the victim item by checking PN_{ig} and PD_{ig} values. Figure 4.1 illustrates this attack for an HPD-based binary PPCF scheme.

In Figure 4.1, an HPD-based P3CF scheme is deployed. The malicious master party has a query vector with 2-group. Notice that the fourth item in the first group and the third item in the second group are the victim items, whose ratings are alienated in the query, these ratings will be derived. After receiving the query vector, the collaborating party calculates required conditional probability values for 2-group and sends it to the master party. Upon receiving conditional probability values, the master party knows that $p(q \mid dislike)$ and $p(q \mid like)$ are calculated for the victim items in the first group and the second group, respectively. Hence, the master party can figure out the value of the victim item relative to q because it does not know the true value of q in the related users' vectors of the collaborating party. For instance, the third item, which is the victim item, in the first user's vector of the collaborating party is opposite of the value of q. This can be verified by checking PN_{11} and PD_{11} of $p(q \mid dislike)$. Since $p(q \mid dislike)$ is only calculated for the third item through all users, PD_{11} reveals if the corresponding item in the user vector is

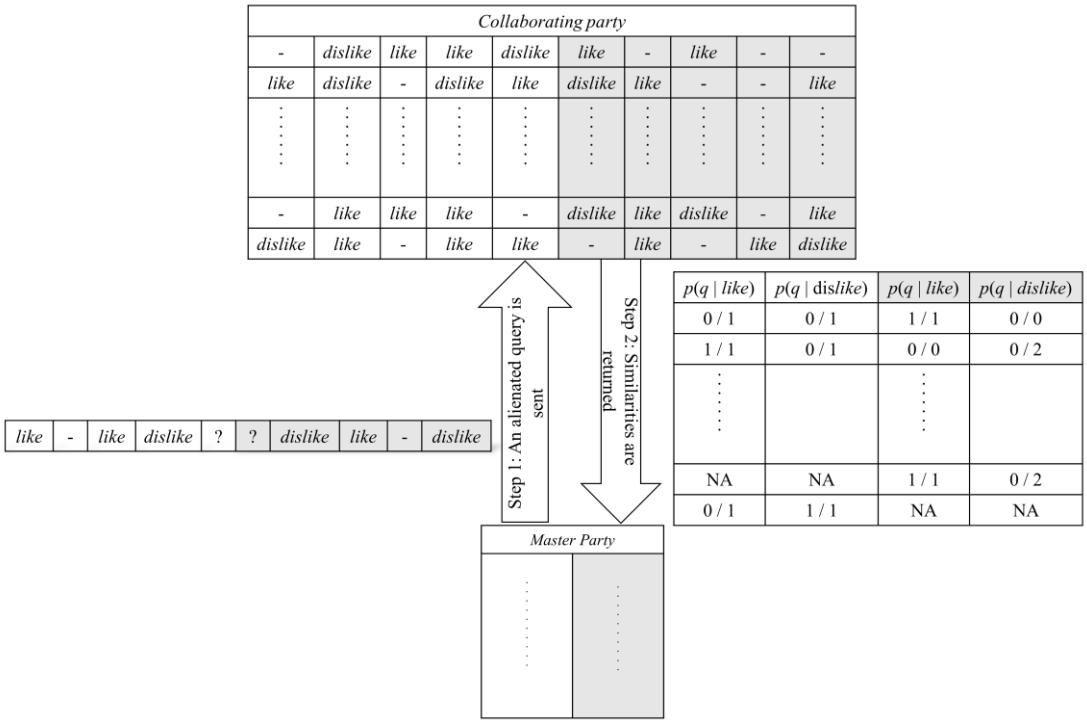


Figure 4.1. *Alienate the victim attack*

rated or unrated. If it is 1, then it rated. If it is 0, then the corresponding item is unrated. Then, the malicious master party must check the value of PN_{11} to derive the rating relative to *q*. If it is 1, then the related item's rating is identical to the value of *q*. Otherwise, it is opposite of the value of *q*. As a result, the master party marks the victim item either opposite or identical to the value of *q*. If the malicious party somehow achieves to learn the value of *q*, the exact rating can be derived. At this point, the auxiliary information can help derive the actual rating, which will be discussed later in this chapter. On the other hand, the exact rating of *q* can be derived in VPD- or VDD-based scenarios. The collaborative parties do not own *q* in VPD- and VDD-based schemes; therefore, they have to calculate all possible conditional probabilities, $p(q = like \mid dislike)$, $p(q = dislike \mid dislike)$, $p(q = like \mid like)$ and $p(q = dislike \mid like)$, to let the master party choose the correct one. Having collected all possible conditional probabilities, the master party can easily figure out the rating. Assume that the master party checks $p(q = dislike \mid dislike)$ returned from the collaborating party to derive the victim item. If PN_{ig} and PD_{ig} are equal and 1, then the victim item in the related user vector is *dislike*. If PN_{ig} is 0 and PD_{ig} is 1, then the victim item is *like*. If PD_{ig} is 0, then it is unrated because PD_{ig} yields 0 when there are no commonly rated items.

4.1.2. Perfect match attack

Perfect match attack exploits the similarity value exchanged between a malicious master and collaborating parties. In this attack, the malicious party initiates a random query and looks for *perfect matches* in similarity values returned from other parties. A perfect match in this context defines a similarity between the active query and any user where all corresponding commonly rated items are identical or opposite to each other. A query with a positive and negative perfect match with two different user vectors is given in Figure 4.2. In a positive perfect match, all corresponding commonly rated items are identical to each other while a negative perfect match occurs when all corresponding commonly rated items are opposite to each other between the query and related user vector. A positive or negative perfect match can be easily detected by a similarity value of 1 or -1, respectively. Such occurrences let the master party infer about the user vector. For example, the malicious master party infers that the user with a positive perfect match holds a vector where each corresponding rating is either identical to the query or unrated. To illustrate, the user with a positive perfect match in Figure 4.2 has both identical ratings and unrated items for the corresponding commonly rated items with the query.

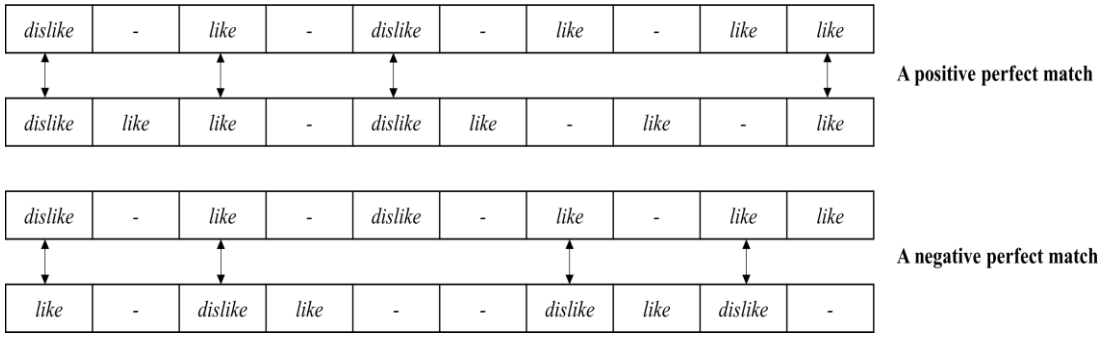


Figure 4.2. *A perfect match*

This attack can be applied in a repeated manner. A new perfect match can be captured in successive queries for a user. In such a case, the master party can infer additional information. Figure 4.3 displays *perfect match* attack with repeated queries. There are three queries with perfect matches. After the master party captures the first perfect, it stores a copy of the first query except the fact that the master party marks corresponding items as *?like* or *?dislike* instead of *like* or *dislike*. ?*like* means that the corresponding item in the user vector is either rated *like* or unrated. Similarly, *?dislike* means that the corresponding item in the user vector is either rated *dislike* or unrated. After the master party captures the second perfect match, it prepares a temporary user

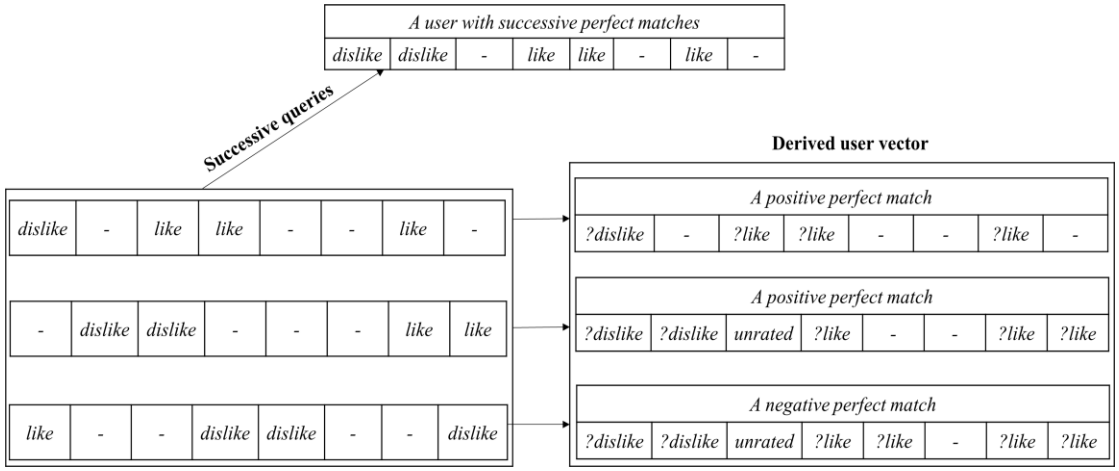


Figure 4.3. *A perfect match attack*

vector that is identical to the second query. This vector is compared with the vector stored for the first perfect match by the master party. If any rating in the first query contradicts with its corresponding rating in the temporary vector, that item is marked as unrated. Notice the third item in the first derived user vector is *?like*; however, it is marked as unrated after this vector is compared with the second temporary user vector. This repeated process is performed for each perfect match to eliminate unrated items.

4.1.3. Acting as an active user attack

Acting as an active user attack exploits temporal changes in the similarity values for repeated queries that differ by only one rating from each other. Multiple queries are needed to derive private data with this attack. A malicious master party acts as AU and sends multiple queries. The malicious master party creates a reference query vector and initiates the recommendation protocol. When similarities are returned from the other parties, the master party stores these similarity values to utilize them in the future. The master party continues to send subsequent queries by only altering one rating at a time with an opposite value. When new similarity values are received, the master party compares the incoming similarity values with the reference, which has been already stored by the master party. If there is an increase in the incoming similarity value, then the correlation (similarity) between the new query and the relevant user increases, which means that the altered rating in the new query is identical to the corresponding rating in the relevant user's vector. If there is a decrease, then they are opposite. If the incoming and reference similarity values are equal, then the corresponding item of the relevant user is unrated.

Figure 4.4 displays *acting as an active user* attack with subsequent queries. First, the similarities for the reference query is stored. Then, the reference query is altered for the first item, and the incoming similarity values are compared with the reference similarity values for each user. If an increase is reported, that item is marked *like*. If a decrease is reported for a user, that item is rated as *dislike*. If similarities remain same, then it is rated unrated as it happens for the second item in Figure 4.4. This process continues after all items are recovered.

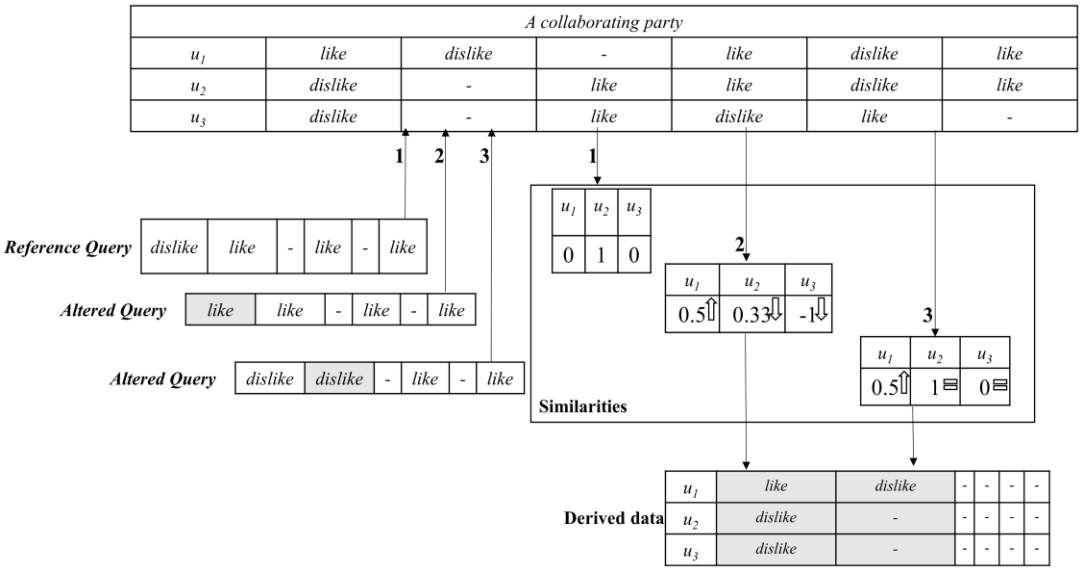


Figure 4.4. *Acting as an active user attack*

Acting as an active user attack is the only attack technique that the targeted distributed schemes (Polat and Du, 2005c; 2008; Kaleli and Polat, 2007a; 2015) in this dissertation consider as a threat and set specific privacy measures. The experiments will be performed to analyze the effects of this attack when privacy measures are taken.

4.1.4. *knn* attack

knn attack targets the selected neighbors in a CF scheme to derive information. This attack is proposed by Calandrino et al. (2011) to disclose information from various online CF services. The main idea in this attack is that the attacker is assumed to have a partial vector of a target user. The attacker creates *k* duplicates of this partially compromised vector and introduces these *k* fake duplicate vectors as new users into a target CF system. Then, the attacker requests a recommendation for one of the *k* fake users. If a CF algorithm utilizes neighborhood, it must select *k* neighbors for an incoming query to produce a recommendation. In such a case, the selected *k* neighbors are expected to

include *k*-1 fake users and the targeted user whose partial vector is known. Since the query and the fake users are identical, the recommendation will be produced from the unknown items in the targeted user's vector.

Figure 4.5 gives an illustration about adding fake users into a CF system. Assume that the attacker has a partial vector of the second vector of the data holder. Lightly shaded part describes the compromised part of the vector. The dark shaded side shows the still confidential data. After fake user vectors are introduced by leaving the confidential part of the vector unrated, the attacker asks for a recommendation for one the fake users. The neighbor selection algorithm might include the partially compromised user's vector and *k*-1 fake user vectors as neighbors of size *k* as shown in the figure. Therefore, the recommendations will be produced from the confidential part of the targeted user vector. The attacker can now figure out which items are rated by the second user of the data holder. However, this attack does not guarantee that *k* neighbors will include the targeted user and *k*-1 fake users. This figure is just an illustration to clarify the understanding of this attack. There might be some other users that can manage to qualify as neighbors of the partially compromised user vector.

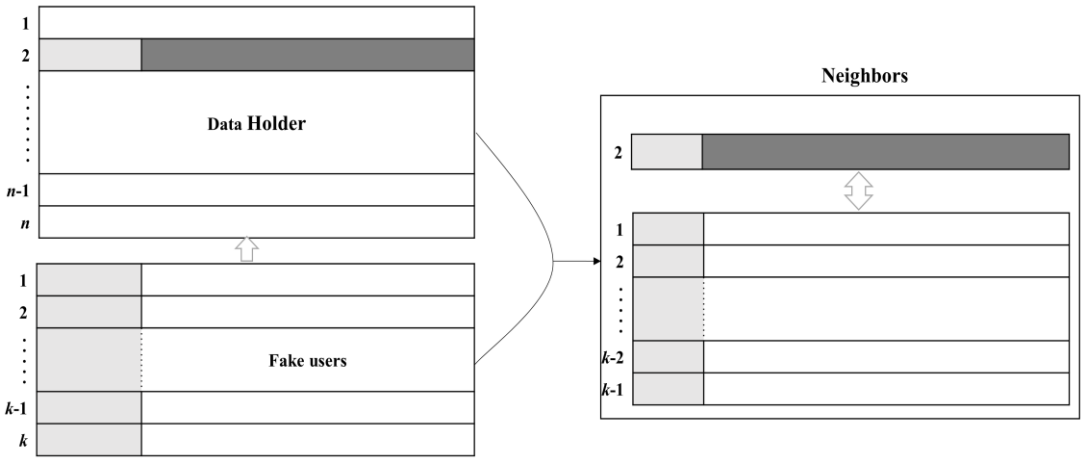


Figure 4.5. *knn attack, introducing fake users*

4.2. The Application of Privacy Attacks on Distributed Binary Schemes

Before delving into the details of how the attacks in this chapter are applied on different binary P3CF and PPDCF schemes, two of these schemes, which are HDD-based threshold (Polat and Du, 2005c; 2008) and NBC-based HDD on PPDCF (Kaleli and Polat, 2007a; 2015), need some clarifications. *Alienate the victim*, *acting as an active user* and *perfect match* attacks exploit the similarity values exchanged between parties. However,

HDD-based threshold scheme (Polat and Du, 2005c; 2008) does not require such an exchange between parties; therefore, these attacks are not applicable for this scheme. On the other hand, these three attacks can only derive whether items of users in collaborating parties are rated or not (second aspect of privacy) when NBC-based binary PPDCF scheme on HDD (Kaleli and Polat, 2007a; 2015) is utilized. The reason behind this phenomenon is that q is hold by each party in HDD; therefore, the master party does not know whether $p(f_u \mid like)$ or $p(f_u \mid dislike)$ is calculated for $f_u = like$ or $f_u = dislike$. Figure 4.6 shows how q resides in NBC-based HDD- and VDD-based schemes (Kaleli and Polat, 2007c; 2015). In HDD-based schemes, collaborating parties hold q of its users, so the master party does not know the value of it. Hence, the malicious master party cannot discover the rating value made for the target item. To overcome this problem, auxiliary information will be used and the details are given in the next section, Chapter 4.3. In contrast to HDD-based schemes, the master party has q in its own institutional data in VDD-based scenarios; thus, the collaborating parties have to calculate all possible conditional probabilities to let the master party choose the correct one based on the value q.

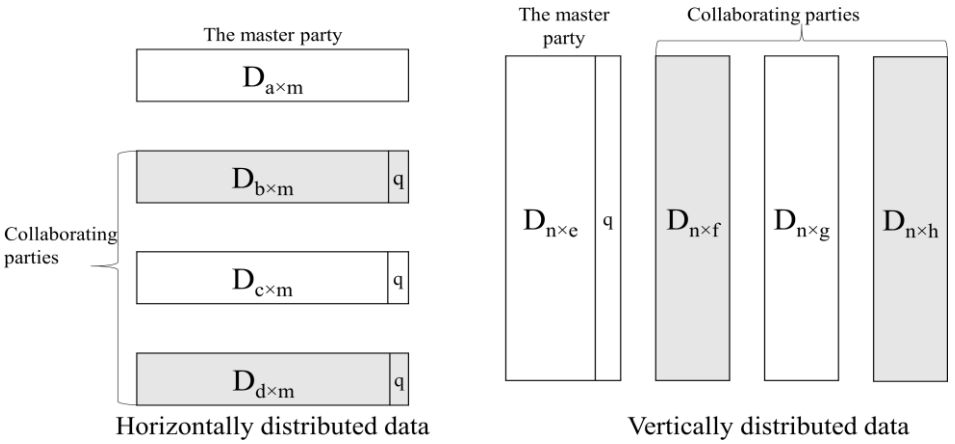


Figure 4.6. *The location of q in NBC-based PPDCF schemes*

Alienate the victim attack singles out an item in an active query so that its ratings can be disclosed. However, this attack is only applicable for NBC-based P3CF and PPDCF schemes (Kaleli and Polat, 2007a; 2015). Parties calculate $p(f_u \mid like)$ and $p(f_u \mid dislike)$ and send these values to the master party by separating related nominator and denominator values. Therefore, the rating for the victim item can be discovered by checking the relevant conditional probability. For example, if the victim item is *like*, the malicious master party must check $p(f_u \mid like)$. Since f_u is determined based on the value

of *q*, the master party should not worry about it. In VPD- and VDD-based schemes (Kaleli and Polat, 2007c; 2015), collaborating parties have to calculate all possible conditional probabilities and let the master party know those probabilities explicitly. Therefore, the master party could exploit the conditional probabilities and discover the rating made for the victim item as discussed in Chapter 4.1.1. In VDD-based schemes, the malicious master party prepares $m - m_{MP}$ number of queries, where m_{MP} is the number of items that the master party holds. A different victim item is picked for each query, and its rating is discovered for all users according to the conditional values returned from other parties.

Perfect match attack exploits similarities values that are 1. This attack can be applied on all P3CF (Polat and Du, 2005c; 2008; Kaleli and Polat, 2007a) and PPDCF (Polat and Du, 2008; Kaleli and Polat, 2015) schemes except threshold-based HDD scheme (Polat and Du, 2005c; 2008). The threshold-based HDD scheme does not require to exchange similarity values, which make *perfect match* attack invalid. On the other hand, *perfect match* needs modification for the NBC-based binary PPDCF on HDD (Kaleli and Polat, 2015) because the master party does not hold *q*. Auxiliary information will be used to derive genuine rating values. *Perfect match* attack must be conducted in a repeated manner to reconstruct data matrices of collaborating parties. As the number of repeated queries increases, the unrated items of collaborating parties start to emerge. High number of repetition of *perfect match* attack will cause the overwhelming majority of unrated items to emerge. The remaining items with either ?*like* or ?*dislike* are marked *like* and *dislike* after the perfect match attack has been repeatedly, respectively. 1000 repetitions with random query vectors will be performed throughout the experiments.

Acting as an active user attack is built upon exploiting the relative change in the similarity values when one of the ratings is reversed in the subsequent query. A reference query is set, and each rated item in the reference query is manipulated to derive their ratings in the users' vectors. After the first reference query is exhausted, new reference queries are created until all of the items are manipulated. After all items are manipulated, and their ratings are discovered, the attack is terminated. Again, this attack will not be applied for threshold-based HDD scheme (Polat and Du, 2005c; 2008).

In *knn* attack, the main idea is to mimic *k* best neighbors with $k-1$ fake users and the user whose partial vector is known. Since NBC-based HDD and VDD PPDCF schemes (Kaleli and Polat, 2015) do not utilize neighborhood, *knn* attack is not applicable to them. VDD-based schemes (Polat and Du, 2005c; 2008) inherently has a history of

ratings for each user because collaborating parties share different ratings for the same set of users. The attacker introduces *k* fake users into the system, which are identical to the user whose ratings will be discovered. This process is repeated until every user is exploited. In HDD-based schemes, the attacker does not have such a luxury to hold a history of any user vector inherently. *knn* will be applied on HDD-based schemes with a strong assumption that the attacker has half of the targeted user vector. This assumption will be repeated for HDD-based schemes during the experiments.

4.3. Exploiting Auxiliary Information

As discussed in the previous section, *alienate the victim*, *perfect match* and *acting as an active user* attacks are applicable to prediction based P3CF and PPDCF schemes (Kaleli and Polat, 2007c; 2015) to derive exact rating values (the first aspect of privacy) if data is vertically distributed between parties. When data is horizontally distributed, the attacker party does not have access to *q*. Thus, an assumption about the value of *q* is needed. This problem will be overcome by auxiliary information.

Auxiliary information is proved to be useful (Demirelli Okkalioglu, Koc and Polat, 2016; Calandrino et al., 2011) to recover information in various scenarios. As repeatedly stated until now in Chapter 4, IMDB was utilized to collect auxiliary information about MLM. This collected information includes average rating, number of votes made for each movie in MLM. This information is exploited in the reconstruction process to disclose private institutional data of collaborating parties. Since the master party does not know the value of *q* in the HDD-based scenario, auxiliary information about *q* might help the malicious master party have an idea about the value of *q*. The master party assumes that the value of *q* for a user might be correlated with the average rating collected from IMDB. The problem with this approach is that *q* is expected to be rated identically to the average rating in IMDB by all users throughout collaborating parties. Hence, a criterion could be adopted to choose which *q* will be queried for the prediction. A query about *q* whose average rating is not decisive such as 6 out of 10 and voted by a few number of IMDB users will not probably serve the initial purpose of having an idea about *q*. It is highly possible that most of the users in collaborating parties hold no rating for *q* whose number of votes in IMDB is very few. In such a case, collaborating parties will not calculate conditional probabilities for those users, and the master party could not derive information about the rating vector. Even if the conditional probability is calculated for a

user, this conditional probability is calculated for *q* whose rating is non-decisive. By non-decisive, it is implied that an average rating such as 6 shows that there is not a consensus over the rating of *q* by IMDB users. Very low and high ratings at least show a consensus toward the unpopularity and popularity of an item, respectively. In this regard, movies with higher than 500,000 number of votes in IMDB are first picked to promote *q*'s that are rated. A list of *q*'s is formed among these movies whose rating is higher than 8.5 or less than 4.0. No movie with a rating less than 4; however, meets these criteria in the collected data set. A random movie (item) is picked as *q* from the list every time a query is dispatched for *alienate the victim* and *acting as an active user* attacks. These attacks are repeated *m* times to derive the ratings of all items. On the other hand, *perfect match* attack can reveal that an item's rating is either unrated or its value relative to *q*. After this attack is repeatedly performed for the NBC-based HDD PPDCF (Kaleli and Polat, 2015), recall that the master party obtains a reconstructed data matrix whose items might contain one of the three possible values, unrated, ?*like* or ?*dislike*. However, the master party can discover whether an item is rated or not by the relevant user by monitoring similarity values. If a collaborating party does not calculate the conditional probability for a user, it means that the relevant user did not rate *q*. Therefore, the master party could also allocate a secondary matrix to map which users rate which items, the second aspect of privacy, by asking as many different *q*'s as for the prediction. Assume that *perfect match* attack has been repeated many times and the master party wants to create a derived matrix. There would be many cells marked such as ?*like*. By exploiting the secondary matrix, the master party could easily figure out whether the relevant cell should be left unrated or marked as *like*. If the relevant cell is marked as rated in the secondary matrix before, then the master party marks the related item as *like*. As a result, it is also very crucial to ask as many different *q*'s as possible to discover the second aspect of privacy. Therefore, *perfect match* attack will utilize random *q*'s in the first half of the repeated queries to discover whether user rate *q* or not while *q*'s selected from the auxiliary information will be utilized in the last half of the repeated queries to promote reliable relative values of *q*.

4.4. Experiments

Experiments have been carried out to see the effects of different privacy parameters on the reconstruction attacks on horizontal and vertical P3CF and PPDCF schemes. Unless otherwise stated δ_{AU} is 0.25, G is 5 and the number of parties is 5. There are a

couple of things to clarify before the details of the experiments are given. In best-*k* HDD P3CF and PPDCF (Polat and Du, 2005c; 2008), the collaborating party permutes neighbors' similarity values and take absolute values of similarities before sending them to the master party. This condition is neglected in the experiments below. The authors give the related privacy analysis about this condition in their study. In terms of *knn* attack, if the number of eligible neighbors is more than *k*, *k* of them are randomly selected. Also, remember that *knn* attacks is associated with the second aspect of privacy while other attacks are associated with the first aspect of privacy.

4.4.1. Effects of varying δ_{AU}

δ_{AU} controls how much fake ratings should be appended to the original rating vector, and it is associated with *d*. In this experiment, δ_{AU} will be varied between 0.125*d*, 0.25*d*, 0.5*d* and 1*d*. The highest value of δ_{AU} is set to 1*d* because appending more ratings than the original vector has is unrealistic. As δ_{AU} gets larger, the perturbed data will become more random. Therefore, the intuition is that the evaluation criteria drop for larger values of δ_{AU} when G and number of parties are fixed at 5. Table 4.1 displays results for HDD- and VDD-based PPDCF scheme while Table 4.2 shows the threshold-based scheme for different attack types. The threshold-based scheme is separated from the rest because its parameter is based on τ. The first columns displaying results with header "No filling" in tables refer a setting where no defense mechanism is applied, a simple CF scheme. DV and RF denote data filling methods. Notice that only *knn* attack is applicable for the threshold-based scheme and *prec* and *rec* are calculated for the second aspect of privacy of it.

Alienate the victim attack relies on a victim item. Notice that DV does not change the status of victim item. Default rating of a vector is appended as fake ratings into the original vector, which does not alter the alienated status of the victim item. Therefore, it is expected that DV will have no effect of preserving private user data when this attack is employed. However, HDD-based schemes utilize auxiliary information since the malicious master party does not have access the rating of *q*. Recall that the attacker has to make an assumption about the rating of *q* using the auxiliary information. Thus, the rating set for *q* will decide the accuracy of this attack rather than DV. Contrary to DV, RF method randomly appends new ratings into unrated cells; therefore, it will definitely deteriorate the evaluation criteria, *prec* and *rec*. Results in Table 4.1 confirms the initial

intuition about DV and RF for NBC-based HDD and VDD PPDCF schemes. It is important to note that this attack performs full recovery in both metrics for the VDD-based scheme when DV is applied even when δ_{AU} gets larger. The rest records a declining trend for larger δ_{AU} values as expected.

Table 4.1. *Effects of varying δ_{AU}*

				δ_{AU}							
				0.125*d*	**0.25*d***	**0.5*d***	**1*d***	**0.125*d***	**0.25*d***	**0.5*d***	**1*d***
			No filling	**DV**				**RF**			
Alienate the victim	NBC HDD	**prec**	0.846	0.483	0.343	0.226	0.148	0.475	0.335	0.214	0.133
		rec	0.452	0.402	0.361	0.300	0.226	0.400	0.357	0.292	0.216
	NBC VDD	**prec**	1.000	1.000	1.000	1.000	1.000	0.611	0.431	0.285	0.169
		rec	1.000	1.000	1.000	1.000	1.000	0.930	0.856	0.753	0.601
Perfect match	NBC HDD	**prec**	0.697	0.699	0.704	0.699	0.693	0.698	0.693	0.699	0.702
		rec	0.053	0.054	0.056	0.056	0.053	0.054	0.051	0.054	0.054
	Best-k	**prec**	0.445	0.448	0.452	0.454	0.445	0.447	0.450	0.449	0.437
		rec	0.931	0.918	0.907	0.885	0.847	0.919	0.907	0.882	0.840
	NBC VDD	**prec**	0.165	0.167	0.170	0.173	0.177	0.167	0.169	0.171	0.175
		rec	0.996	0.995	0.995	0.994	0.993	0.994	0.993	0.989	0.981
	Case-All	**prec**	0.455	0.455	0.456	0.451	0.439	0.458	0.458	0.458	0.450
		rec	0.930	0.921	0.913	0.897	0.867	0.920	0.910	0.889	0.854
	Case-Split	**prec**	0.356	0.358	0.362	0.372	0.375	0.364	0.371	0.380	0.387
		rec	0.957	0.949	0.941	0.924	0.896	0.947	0.938	0.918	0.883
Acting as an active user	NBC HDD	**prec**	0.852	0.420	0.283	0.191	0.128	0.409	0.272	0.181	0.117
		rec	0.471	0.397	0.353	0.324	0.286	0.401	0.354	0.346	0.272
	Best-k	**prec**	1.000	0.271	0.192	0.128	0.091	0.277	0.190	0.121	0.094
		rec	1.000	0.983	0.966	0.943	0.921	0.985	0.968	0.940	0.924
	NBC VDD	**prec**	1.000	0.520	0.383	0.255	0.180	0.525	0.366	0.237	0.145
		rec	1.000	0.933	0.894	0.844	0.785	0.911	0.859	0.783	0.675
	Case-All	**prec**	1.000	0.334	0.210	0.135	0.096	0.321	0.212	0.142	0.096
		rec	1.000	0.988	0.965	0.922	0.872	0.987	0.965	0.927	0.870
	Case-Split	**prec**	1.000	0.325	0.216	0.143	0.094	0.334	0.217	0.137	0.099
		rec	1.000	0.988	0.967	0.927	0.869	0.988	0.968	0.923	0.874
knn	Best-k	**prec**	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047
		rec	0.426	0.426	0.426	0.426	0.426	0.425	0.426	0.426	0.425
	Case-All	**prec**	0.058	0.058	0.058	0.058	0.059	0.058	0.058	0.059	0.060
		rec	0.449	0.448	0.447	0.448	0.449	0.448	0.446	0.448	0.442
	Case-Split	**prec**	0.071	0.070	0.069	0.069	0.067	0.070	0.070	0.070	0.070
		rec	0.520	0.510	0.497	0.478	0.442	0.516	0.512	0.505	0.493

There are five different PPDCF schemes that *perfect match* attack can be applied as seen in Table 4.1. The general inference made from the results is that results do not show dramatic declining or increasing trends as *alienate the victim* attack in the previous paragraph. In terms of the NBC-based HDD PPDCF scheme, *prec* results change between 0.693 and 0.704, which is the best among all of the schemes. However, *rec* is very low and fluctuating between 0.051 and 0.056. This means that derived ratings from *perfect match* attack are reliable (high precision), but the amount of correctly derived ratings corresponds to the small fraction of original ratings (low recall). On the other hand, very high *rec* values around 0.900 are very common for the rest of the schemes. For example, *perfect match* attack records very remarkable *rec* values around 0.990 for different δ_{AU} values when the NBC-based vertical PPDCF scheme is employed although *prec* values are lower than the other schemes. Low precision means that derived ratings contain many unrelated ratings in size while high recall means that high percentage of original ratings are indeed recovered. *Perfect match* attack demonstrates promising results in terms of *rec* for best-*k*, Case-All and Case-Split as well.

Acting as an active user attack should perform full *prec* and *rec* results when no privacy measures are taken. However, the NBC-based horizontal PPDCF scheme utilizes auxiliary information to predict the value of *q*; therefore, *prec* and *rec* are lower than 1.000 for this scheme when HRI is not applied. When δ_{AU} increases, there is a declining trend due to HRI. Repetitive queries should differ from each other with only by one cell; nonetheless, HRI with higher δ_{AU} possibly makes each subsequent query differ dramatically. As a result, this attack yields very high *rec* results and lower *prec* for all schemes except NBC-based horizontal one.

Results for *knn* attack are given in both Table 4.1 and 4.2. The latter table includes results for threshold-based horizontal PPDCF scheme since the parameter is different for this scheme. This scheme selects neighborhood based on τ and it does not apply HRI and therefore separated from the rest. Results show stable trends for all schemes in *knn* attack although measures are tightened. If there are more than *k* users who are eligible to become neighbors, *k* of them are randomly picked. With increasing values of privacy measures, there is not a significant change in the results so one can argue that there are always some unintended neighbors. Very low *prec* results describe that most of the recovered ratings do not match with the original ratings. In terms of *rec*, *knn* attack records moderate results.

Table 4.2. *Effects of varying δ_{AU} on the HDD-based threshold scheme*

		τ	**No threshold**	**0.125**	**0.25**	**0.75**
knn	Threshold	**prec**	0.032	0.032	0.031	0.032
		rec	0.327	0.325	0.303	0.285

However, the drawback of this attack is that *prec* values are too low to be considered effective for different PPDCF schemes for varying δ_{AU}.

4.4.2. Effects of varying G

Parties could divide ratings of its users into a different number of groups, G, as discussed. For each group, RRT is applied independently. The problem with the one-group scheme is that any disclosure of an item reveals whole rating vector due to RRT because ratings are either reversed or preserved. Thus, a disclosure of an item in a multi-group vector only reveals ratings of other items in the same group. This experiment investigates the effects of G on the reconstruction results against different attacks given in this study. G is varied between 1, 3, 5, 10 and 20. It is worth mentioning that G used in this experiment defines the number of groups for a whole user vector; for example, if G is 5 for a vertically distributed user vector, each party has one-group for their part in a 5-party scheme. There is also one more point to clarify with VDD. When G is not a factor of the number of parties, the actual total number of groups could exceed G. For example, when G is 3 and the number of parties are 5, the actual G becomes 7 as shown in Figure 4.7.

Remember that HRI protocol appends ratings into AU's query up to δ_{AU} which is associated with *d*. An AU's query is filled with an average of $\delta_{AU} / 2$ fake ratings which contribute to privacy. However, it is hypothesized that increasing G values under a constant δ_{AU} will help reconstruction results. When G is increased, the possibility of each group to be perturbed by HRI decreases. Assume that $G = m$, where m is the number of items, interim calculations are made for each group, and none of the groups of the geniune rated items (each item constitutes a group) are manipulated by HRI. Since the master party knows true AU's query, it can easily capture true interim results. As a result, the hypothesis is that reconstruction results will improve as G increases. Table 4.3 shows the experimental results.

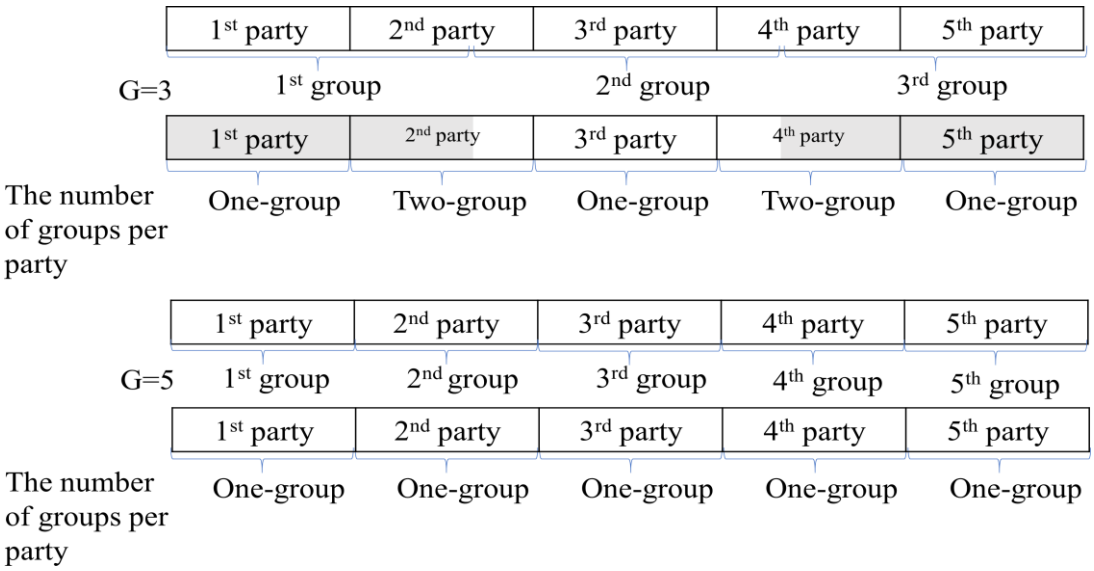


Figure 4.7. *Number of groups with VDD*

Alienate the victim attack singles out a victim item from the rest to derive its rating. Results show that *prec* and *rec* metrics improve for larger G values. As G increases, it is less probable that a victim item's group will be invaded by fake ratings introduced due to HRI. Therefore, increasing values of G yields better results regardless of the filling methods, DV or RF. In NBC-based horizontal PPDCF, *prec* value starts around 0.120 when G is 1 and hikes up to around 0.600 for both of the filling methods when G is 20. *Rec* values also demonstrate gradual increase up to around 0.430. The gradual increasing trends are also very clear with the NBC-based vertical PPDCF scheme with two exceptions. *Prec* and *rec* are 1.000 when G is 1 and 5 and the filling method is DV for the NBC-based vertical PPDCF scheme. DV appends the dominant rating in the vector, and it is clear that this does not alter the isolated status of the victim item when G is 1. Similarly, when G is 5, each party has exactly 1-group since the total number of groups is 5. When RF is examined, the trends is very clear and numbers follow an increasing trend as G increases. In vertical schemes, results are becoming very promising with especially large G values.

Perfect match attack creates random queries and looks for positively or negatively perfectly matched users. As G increases, AU's vector will be split into smaller parts. This might cause more perfect matches to be captured. This attack records slight increases for the NBC-based HDD scheme in *prec* and *rec*. However, *rec* values change between 0.033 and 0.060 for different G values and filling methods, DV and RF. Such *rec* values are too

low. On the other hand, *prec* change between 0.680 and 0.708 with a generally increasing trend. When the best-*k* scheme is applied, *rec* consistently rises up to 1.000 for DV and RF. This means that all of the original ratings can be derived when G is close to 20. However, *prec* reaches its peek when G is 5. When G approaches 20, this attack

Table 4.3. *Effects of varying G*

			G									
			1	**3**	**5**	**10**	**20**	**1**	**3**	**5**	**10**	**20**
					DV					**RF**		
Alienate the victim	**NBC HDD**	**prec**	0.126	0.254	0.343	0.467	0.600	0.113	0.241	0.335	0.472	0.595
		rec	0.204	0.318	0.361	0.398	0.428	0.194	0.315	0.357	0.400	0.427
	NBC VDD	**prec**	1.000	0.923	1.000	0.763	0.823	0.437	0.495	0.431	0.594	0.724
		rec	1.000	0.990	1.000	0.959	0.970	0.861	0.883	0.856	0.915	0.942
Perfect match	**NBC HDD**	**prec**	0.684	0.696	0.704	0.705	0.701	0.680	0.695	0.693	0.693	0.708
		rec	0.033	0.053	0.056	0.058	0.060	0.033	0.050	0.051	0.056	0.058
	Best-k	**prec**	0.127	0.386	0.452	0.352	0.253	0.127	0.383	0.450	0.351	0.253
		rec	0.353	0.753	0.907	0.990	1.000	0.351	0.751	0.907	0.989	1.000
	NBC VDD	**prec**	0.169	0.149	0.170	0.130	0.113	0.169	0.149	0.169	0.129	0.113
		rec	0.995	0.997	0.995	1.000	1.000	0.993	0.996	0.993	1.000	1.000
	Case-All	**prec**	0.231	0.300	0.456	0.351	0.251	0.230	0.306	0.458	0.357	0.257
		rec	0.026	0.188	0.913	0.991	1.000	0.026	0.188	0.910	0.991	1.000
	Case-Split	**prec**	0.283	0.064	0.362	0.261	0.195	0.281	0.064	0.371	0.266	0.199
		rec	0.007	0.038	0.941	0.995	1.000	0.008	0.038	0.938	0.995	1.000
Acting as an active user	**NBC HDD**	**prec**	0.113	0.211	0.283	0.407	0.551	0.104	0.206	0.272	0.404	0.559
		rec	0.279	0.345	0.353	0.392	0.423	0.265	0.358	0.354	0.397	0.424
	Best-k	**prec**	0.078	0.146	0.192	0.261	0.401	0.087	0.144	0.190	0.283	0.370
		rec	0.914	0.950	0.966	0.982	0.992	0.922	0.952	0.968	0.984	0.991
	NBC VDD	**prec**	0.385	0.432	0.383	0.524	0.681	0.371	0.366	0.526	0.526	0.689
		rec	0.904	0.907	0.894	0.933	0.961	0.865	0.874	0.859	0.911	0.952
	Case-All	**prec**	0.223	0.247	0.210	0.314	0.460	0.230	0.306	0.212	0.357	0.257
		rec	0.968	0.974	0.965	0.985	0.995	0.026	0.188	0.965	0.991	1.000
	Case-Split	**prec**	0.206	0.251	0.216	0.334	0.466	0.210	0.258	0.217	0.316	0.477
		rec	0.964	0.974	0.967	0.987	0.995	0.965	0.976	0.968	0.986	0.995
knn	**Best-k**	**prec**	0.047	0.047	0.047	0.047	0.047	0.048	0.046	0.047	0.047	0.047
		rec	0.427	0.425	0.426	0.426	0.426	0.425	0.427	0.426	0.427	0.426
	Case-All	**prec**	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058	0.058
		rec	0.447	0.449	0.447	0.449	0.448	0.447	0.447	0.446	0.447	0.448
	Case-Split	**prec**	0.069	0.069	0.069	0.069	0.070	0.070	0.070	0.070	0.070	0.070
		rec	0.498	0.498	0.498	0.498	0.498	0.512	0.513	0.513	0.513	0.513

demonstrates a declining trend in terms of *prec* for best-*k*. The trend for *prec* is similar to the VDD-based schemes as well. It reaches its peak at G = 5 and a decline follows toward G is 20. Increasing G toward 20 promotes *rec* up to 1.000 for VDD-based schemes in a similar way it occurs for best-*k*. To sum up, increasing values of G promotes especially *rec* for PPDCF schemes except for the NBC-based HDD scheme.

Acting as an active user attack monitors temporal changes between subsequent queries manipulated by only one cell. Therefore, HRI protocol should not append fake ratings into the groups of the manipulated item for this attack to be successful. As G increases, this possibility increases; thus, it is expected that the results will get better. With HDD-based schemes (NBC and best-*k*), increase in both metrics is obvious and consistent. For example, when G is 1, *prec* is 0.113 and 0.104 for DV and RF, respectively. When G approaches to 20, *prec* consistently climbs up to 0.551 and 0.559 for DV and RF, respectively. In the VDD-based schemes (NBC, Case-All and Case-Split), this attack demonstrates increasing trends for larger G values. When G is 3, *prec* and *rec* results are always better than the case where G is 5. The reason behind this phenomenon is that the actual number of groups is greater when G is 3 than G is 5 when the number of parties is 5 as shown in Figure 4.7.

Results for *knn* attack are very stable for varying group numbers. Remember that these numbers are calculated for the second aspect of privacy. Although *rec* values can be considered reliable, *prec* values are very low. Low *prec* values indicate that derived item list of the attacker contains a high volume of unrelated items. Zhang, Ford and Makedon (2006) indicate that precision is more important for an attacker. As a result, *knn* attack follows a stable trend as it did in the previous experiment. However, very low values of *prec* render this attack impractical.

4.4.3. Effects of varying number of parties

The above experiments are performed with 5 different parties. However, the number of parties could vary. In this experiment, the number of parties are varied between 2, 3, 5 and 10. Since multi-party, PPDCF, schemes (Polat and Du, 2008; Kaleli and Polat, 2015) are extended upon two-party, P3CF, schemes (Polat and Du, 2005c; 2008; Kaleli and Polat, 2007a), P3CF schemes are not included earlier experiments in this section. They will be now demonstrated in this experiment by starting the number of parties from

2 to 10. Results are given in Table 4.4 and 4.5, and the latter table includes threshold-based scheme.

Results for the NBC-based HDD scheme when *alienate the victim* attack is applied varies between 0.327 and 0.343 for *prec* and 0.348 and 0.361 for *rec* with DV. Although results are in similar ranges, a marginal increase is observed through the 5-party scheme.

Table 4.4. *Effects of varying number of parties*

			Number of parties							
			2	**3**	**5**	**10**	**2**	**3**	**5**	**10**
			DV				**RF**			
Alienate the victim	NBC HDD	**prec**	0.327	0.335	0.343	0.333	0.318	0.323	0.335	0.322
		rec	0.348	0.359	0.361	0.354	0.347	0.355	0.357	0.352
	NBC VDD	**prec**	0.650	0.777	1.000	1.000	0.418	0.466	0.431	0.623
		rec	0.934	0.957	1.000	1.000	0.845	0.867	0.856	0.927
Perfect match	NBC HDD	**prec**	0.716	0.711	0.704	0.696	0.721	0.716	0.693	0.690
		rec	0.072	0.067	0.056	0.041	0.075	0.068	0.051	0.041
	Best-k	**prec**	0.467	0.463	0.452	0.465	0.465	0.461	0.450	0.465
		rec	0.901	0.908	0.907	0.902	0.899	0.907	0.907	0.901
	NBC VDD	**prec**	0.177	0.161	0.170	0.120	0.177	0.161	0.169	0.119
		rec	0.994	0.996	0.995	1.000	0.991	0.994	0.993	1.000
	Case-All	**prec**	0.496	0.453	0.456	0.074	0.498	0.462	0.458	0.076
		rec	0.799	0.582	0.913	0.125	0.794	0.580	0.910	0.124
	Case-Split	**prec**	0.406	0.322	0.362	0.028	0.412	0.332	0.371	0.028
		rec	0.757	0.545	0.941	0.051	0.753	0.542	0.938	0.050
Acting as an active user	NBC HDD	**prec**	0.270	0.277	0.283	0.277	0.265	0.274	0.272	0.266
		rec	0.355	0.366	0.383	0.371	0.355	0.361	0.354	0.342
	Best-k	**prec**	0.181	0.182	0.192	0.192	0.164	0.173	0.190	0.175
		rec	0.966	0.965	0.966	0.967	0.961	0.963	0.968	0.964
	NBC VDD	**prec**	0.372	0.518	0.383	0.542	0.353	0.386	0.366	0.553
		rec	0.877	0.934	0.894	0.940	0.836	0.856	0.859	0.924
	Case-All	**prec**	0.207	0.237	0.210	0.344	0.498	0.462	0.212	0.336
		rec	0.963	0.970	0.965	0.989	0.794	0.580	0.965	0.988
	Case-Split	**prec**	0.196	0.235	0.216	0.342	0.208	0.246	0.217	0.338
		rec	0.958	0.969	0.967	0.989	0.962	0.970	0.968	0.988
knn	Best-k	**prec**	0.049	0.048	0.047	0.046	0.048	0.048	0.047	0.046
		rec	0.423	0.426	0.426	0.430	0.424	0.425	0.426	0.430
	Case-All	**prec**	0.064	0.065	0.058	0.050	0.064	0.065	0.058	0.050
		rec	0.443	0.464	0.447	0.436	0.443	0.465	0.446	0.436
	Case-Split	**prec**	0.081	0.075	0.069	0.070	0.082	0.076	0.070	0.071
		rec	0.490	0.495	0.497	0.490	0.500	0.506	0.512	0.505

This attack performs very similar results when RF is utilized. The way of filling unrated items differs when the NBC-based VDD scheme is applied. When the filling method associated with HRI is DV, results are promising for varying number of parties. *Prec* and *rec* are 1.000 when the number of parties is 5 and 10. The reason for this phenomenon is that the actual number of groups per party is 1 when the number of parties is 5 and 10. When data filling method is RF, the reconstruction results increase toward 10-party scheme. Especially, *prec* and *rec* are 0.623 and 0.927, respectively.

When *perfect match* attack is utilized for varying number of parties, *prec* values remain relatively steady from 2- to 10-party in the NBC-based and best-*k* HDD schemes for both of the filling methods, DV and RF. On the hand, *rec* follows a falling trend and very low for the NBC-based HDD scheme. At this point, recall that the NBC-based HDD scheme utilizes auxiliary information. On the other hand, *rec* is almost above 0.900 and remains almost stable for all cases in the best-*k* scheme. For all of VDD-based schemes, this attack generally performs stable *prec* results except a steep decline for 10-party cases in Case-All and Case-Split. In terms of the NBC-based VDD scheme, *rec* results are very promising.

Acting as an active user attack remains relatively stable with marginal changes for HDD-based schemes (NBC and best-*k*) for both metrics. On the other hand, both metric fluctuate for the NBC-based VDD scheme while there is an increasing trend for a larger number of parties in Case-All and Case Split.

knn attack remains almost stable for varying number of parties for all schemes in terms of *prec* and *rec* as given in Table 4.4 and 4.5. One can conclude that varying number of parties does not affect the reconstruction results of the attack. Once again, note that *prec* is very low.

4.4.4. Effects of privacy measure for extreme cases

As discussed in Chapter 2.7, two extreme cases might occur that could disclose confidential information. The first case is related to users who did not rate *q*. A malicious

Table 4.5. *Effects of varying parties on HDD-based threshold scheme*

		Number of parties	**2**	**3**	**5**	**10**
knn	Threshold	**prec**	0.032	0.032	0.032	0.031
		rec	0.285	0.285	0.285	0.287

master party could disclose the users rating *q*, which is a violation of the second aspect of privacy. NBC-based schemes (Kaleli and Polat, 2015) offer to fill unrated *q*'s of some users with a random percentage up to d_{set} to let unrelated users participate in the PPCF algorithm for the related party. Notice that this case is called PPP in Chapter 2.7. In the second extreme case, users might rate *q* but do not have any common ratings with AU. A malicious master party could infer that the related user did not rate any rating that the AU rate. The authors (Kaleli and Polat, 2015) propose to use HRI on the related user vector where no corresponding entries match to fill some of them. This privacy measure is called PPR in Chapter 2.7. Note that the first extreme case is not valid for the NBC-based VDD scheme because parties do not have the part of the vector to which *q* belongs. Parties calculate similarity values for all possible values of *q*. Table 4.6 and 4.7 display the results for the NBC-based HDD and VDD schemes, respectively. Reference column in the tables refers to the case where no extreme privacy measures are applied.

Since PPP adds some users who did not rate *q* into prediction process and PPR perturbs users' vectors who have a rating for *q* yet no common ratings with the AU, a declining trend is expected. When PPP and PPR are applied, the decline in evaluation metrics are not very apparent considering the reference case. For example, the highest decline in *prec* is 0.008 and 0.007 for DV and RF are applied, respectively, when *alienate the victim* attack is applied. Likewise, the decline for other attack types when extra privacy measures are applied are not prominent. Since the NBC-based HDD scheme utilizes auxiliary information to determine *q*, there is already an inherited randomness in the process; therefore, the integrated privacy measures might not be effective in a dramatic extent. Even, evaluation metrics increase with RF when both PPP and PPR are applied for *perfect match* and *acting as an active user* attacks.

The NBC-based VDD scheme can only employ PPR as discussed. Since some user vectors are perturbed by a similar method to HRI, randomness increases, and it would negatively affect the reconstruction results. Table 4.7 displays the results and decline is reported for all attack types in terms of *prec*. However, the degree of the decline is not very dramatic for *prec*. For example, *prec* is 1.000 and 0.431 with the reference case when *alienate the victim* attack is utilized with DV and RF, respectively. When PPR is added by the parties as an extra measure, these numbers fall to 0.973 and 0.426. The only credible decline is recorded for *acting as an active user* attack, *prec* fell from 0.383 and

0.366 to 0.313 and 0.306 for DV and RF, respectively. On the other hand, *rec* values remain almost stable when PPR is applied.

As a result, PPP and PPR, which are applied to provide privacy against two extreme cases, do not offer privacy as much as expected. The decline in *prec* is very limited while *rec* remains almost stable for both HDD- and VDD-based PPDCF schemes.

Table 4.6. *Effects privacy measures against extreme cases, the NBC-based HDD scheme*

			Reference	PPP	PPR	Both	Reference	PPP	PPR	Both
				DV				RF		
Alienate the victim	NBC HDD	**prec**	0.343	0.336	0.332	0.335	0.335	0.336	0.331	0.328
		rec	0.361	0.358	0.356	0.357	0.357	0.358	0.358	0.357
Perfect match	NBC HDD	**prec**	0.704	0.699	0.693	0.694	0.693	0.703	0.694	0.697
		rec	0.056	0.053	0.057	0.053	0.051	0.056	0.057	0.055
Acting as an active user	NBC HDD	**prec**	0.283	0.285	0.287	0.278	0.272	0.289	0.277	0.282
		rec	0.383	0.381	0.368	0.373	0.354	0.388	0.375	0.370

Table 4.7. *Effects privacy measures against extreme cases, the NBC-based VDD scheme*

			Reference	PPR	Reference	PPR
			DV		RF	
Alienate the victim	NBC VDD	**prec**	1.000	0.973	0.431	0.426
		rec	1.000	1.000	0.856	0.860
Perfect match	NBC VDD	**prec**	0.170	0.160	0.169	0.159
		rec	0.995	0.995	0.993	0.993
Acting as an active user	NBC VDD	**prec**	0.383	0.313	0.366	0.306
		rec	0.894	0.884	0.859	0.859

4.5. Conclusion

In this part, institutional privacy offered in P3CF and PPDCF schemes is examined. Four different attacks have been discussed and their success to derive original institutional data is experimentally tested. *Alienate the victim* and *perfect match* attacks that exploit exchanged similarity values are proposed. *Acting as an active user* and *knn* attacks are well-known attacks, and they are also implemented in this chapter. Throughout the experiments, different parameters are controlled to see how they affect the reconstruction results.

Alienate the victim attack records a declining trend when δ_{AU} is increased to control how much of the AU's vector should be appended with fake ratings. Recall that HRI protocol has two different kinds of filling methods, DV and RF while filling unrated items

of AU. However, when DV is utilized for NBC-based VDD scheme, this attack achieves *prec* and *rec* values of 1.000 for all values of δ_{AU} because DV does not hinder the status of the victim item due to the fact that unrated items are filled with the dominating rating. The second experimental setting tests how varying G affects the reconstruction. An increasing trend for *prec* and *rec* is recorded for larger G values. Such results contradict with an initial thought that larger G should decrease the reconstruction results; however, when G is increased, it is probable that the alienated status of victim item remains unaltered. Nonetheless, there is an exception with NBC-based VDD scheme with DV due to the number of groups corresponding to each party if G is not a factor of the number of parties. In the third experimental setting, the number of parties is tested. The most important point is *prec* and *rec* results that are 1.000 when the number of parties 5 and 10 when partitioning is vertical and the filling method is DV. The reason is that the number of groups for each party is 1 with 5- and 10-party schemes; therefore, the victim item remains alienated. As a result, *alienate the victim* attack achieves very good results especially with DV, and it could become very destructive to derive private institutional data. The experiments show that the relation between G per party is very important to avoid this attack.

When *perfect match* attack is employed for different control parameters in the experiments, the general trend of this attack shows a stable trend. This attack usually performs very high *rec* results for all PPDCF schemes except NBC-based HDD scheme. It should be noted that attacks applied on this scheme have to utilize auxiliary information to assign the value of *q*. The use of auxiliary information might cause such low *rec* results for NBC-based HDD algorithm. On the other hand, *prec* results recorded for NBC-based VDD scheme by this attack are not promising when compared to the success of *alienate the victim* attack on this algorithm. *Perfect match* attack clearly shows promising *rec* results; however, *prec* is usually considered more important for the adversaries (Zhang, Ford and Makedon, 2006) because adversaries would be more interested in the volume of accurately derived ratings. An adversary would prefer higher *prec* and lower *rec* pair rather than lower *prec* and higher *rec* pair.

Acting as an active user attack declines for larger δ_{AU} values in terms of *prec* and *rec* values for all PPDCF schemes. This declining trend is in line with the expectation because larger δ_{AU} values append more fake ratings into the AU's query. Such fake ratings alter the manipulated query which is different from the reference query by only one cell.

When G is varied, both metrics increase for larger G. Similar to *alienate the victim* attack, increasing G values create more groups and it would be less possible to append fake ratings into the group where manipulated item belongs to. The third experiment tests the effect of the number of parties. This attack does not show dramatic fluctuation for varying number of parties.

The last attack, *knn*, derives whether an item is rated or not, the second aspect of privacy. Throughout all experiments, *prec* and *rec* results almost remain stable for all different experimental settings when *knn* attack is employed. An important drawback with this attack is the low *prec* values although *rec* values could be considered as moderate.

Apart from the three experimental settings that deal with parameters of HRI and number of parties involved in the PPCF process, participating parties other than the master party could take extra privacy measure to prevent from possible extreme situations. When these extreme privacy measures are introduced, the decline in reconstruction results can be regarded as too marginal to be considered effective for NBC-based PPDCF schemes.

5. DERIVING PRIVATE DATA FROM P2P BINARY PPCF SCHEMES

Although server-based schemes, which are central, partitioned or distributed, have privacy measures to preserve the privacy of individual or institutional data, peers can come together and establish their P2P network for CF purposes without a data holder. A P2P binary PPCF scheme studied by Kaleli and Polat (2010) is targeted in this chapter. This is also an NBC-based prediction scheme where an AP perturbs his or her vector before asking for a prediction to preserve the privacy of her rating vector from other peers. On the other hand, peers can also apply privacy measures to prevent data disclose while collaborating for a prediction. Details of this scheme are given in Chapter 2.7, where preliminaries are introduced. The objective of this chapter is to derive peers' rating vectors when an AP acts maliciously. Attacks introduced in Chapter 4 to derive original ratings will be applied for the P2P binary PPCF scheme too. Each party holds original user data, and distributed PPCF protocols are adapted to preserve the privacy of institutional data from other parties. However, the objective in P2P PPCF is to preserve the individual privacy of peers. Therefore, the main distinction between the previous and this chapter is that the first aims to derive private institutional data while the latter aims to derive private individual peer's data. Since the same set of attacks will be applied on P2P binary PPCF scheme, they will not be introduced here. Their application on the P2P binary PPCF scheme (Kaleli and Polat, 2010) will be discussed in the following subsection.

5.1. The Application of Attacks on P2P Binary PPCF Schemes

P2P PPCF schemes are similar to HDD-based PPCF schemes in nature because each peer has its own rating vector. In P2P setting with privacy, each peer could be considered as an independent party and q is held by each peer. Thus, the only difference between HDD-based and P2P scenarios is that a party has many original user vectors while each peer has her own vector, respectively. Since q does not reside in AP's vector, AP is not able to know whether $p(q=like \mid c_j)$ or $p(q=dislike \mid c_j)$ where $c_j \in \{like, dislike\}$ is returned from peers. Similar to the attacks applied on the NBC-based HDD scheme (Kaleli and Polat, 2015), the attacks in this chapter need auxiliary information as well to estimate the value of q residing at peers' sites while deriving peers' individual data (the first aspect of privacy).

Alienate the victim attack can be employed for the P2P binary scheme by Kaleli and Polat (2010) in a repeated manner by picking a unique victim item each time a new query is dispatched so that each item is set victim once to derive its possible rating. Therefore, this attack is repeated m times to attempt to derive all ratings from other peers. A malicious AP prepares a query with one victim item and asks for a prediction. *Perfect match* attack is also suitable for this scheme. As discussed in Chapter 4, this attack recovers private data by controlling similarity values that are either 1 or -1, perfect matches. This attack is also applied in $2 \times m$ times to increase the number of repetitions. In the first m successive queries, unique q values for each item are queried to create a mapping of rated items. The second m queries, q's selected from auxiliary information are queried. The third attack, *acting as an active user*, can also be applied by manipulating a rating at a time from the reference query. Therefore, this attack needs to be repeated m times to try to derive all items.

The targeted scheme (Kaleli and Polat, 2010) in this chapter is an NBC-based scheme that provides predictions between peers. Since this scheme neither utilizes neighboring approach nor provides a recommendation, top-N results, *knn* attack will not be included in the attacks for the P2P binary PPCF.

5.2. Exploiting Auxiliary Information

AP needs to know the rating made for q by peers to derive peers' ratings; however, each peer holds q so AP is not aware of the value of q. On the other hand, the attacks in this dissertation given for PPDCF schemes, which are also applicable for the P2P scheme given in this chapter, cannot derive actual rating values if AP does not know q. The NBC-based P2P binary PPCF scheme (Kaleli and Polat, 2010) does not require the value of q to calculate the final probabilities. Thus, AP needs auxiliary information to make an assumption about the value of q that peers have for all attack types. AP uses the same set of auxiliary information that has been used in Chapter 4 and the way how the value of q is estimated is identical as well. Movies with higher than 500,000 votes whose ratings are greater than 8.5 or less than 4.0 are picked as q. *Alienate the victim* and *acting as an active user* attacks have utilized a different q each time this attack is repeated to derive the rating of each item in peers' vectors. Hence, these two attacks have been repeated m times to derive the rating of every item. *Perfect match* attack is applied with a little modification because it can reveal whether an item's rating is either unrated or its value relative to q as

discussed previously. *Perfect match* attack is performed in two-steps. First, each item is queried to create a mapping of which peer rated which item, the second aspect of privacy. Then, the second step is performed for *m* times with movies which can be nominated as *q* similar to other two attacks. This way of applying this attack in two-steps is different from its application for HDD-based schemes in the previous chapter. It is applied 1000 times in Chapter 4 for HDD-based schemes, although the half of them is to create a mapping of rated items and the other half is to use *q*'s observed from auxiliary information.

5.3. Experiments

Experiments have been carried out with different control parameters that are δ_{AP}, the filling method, G, and peer privacy measures, PPP and PPR. Unless otherwise stated, δ_{AP} is $0.25d$, G is 5 and peer privacy measures are not active. The density of each query dispatched from AP is prepared is associated with *d*. Each experiment is repeated 5 times and their averages are taken.

5.3.1. *Effects of δ_{AP} and filling methods*

AP utilizes HRI protocol to hide ratings by appending up to δ_{AP} density. Note that δ_{AP} value is associated with *d* and ratings are appended based on two different filling methods, RF and DV. It is obvious that introducing δ_{AP} should decrease the reconstruction metrics because some unintended ratings are appended to AP's rating vector. Reconstruction results for *acting as an active user* attack could be affected by increasing δ_{AP} values regardless of which filling method is utilized. This attack relies on subsequent queries that differ by only one item rating at a time. When δ_{AP} and a filling method are utilized for a query, the next queries will be much more different from the intended one that is expected to differ only one item rating. Therefore, the intuition is that altering each departing query for participating peers by δ_{AP} and the filling method will definitely diminish the results. The resilience of *alienate the victim* attack against privacy measures, δ_{AP}, and the filling method, could depend on the filling method that is used. This attack relies on singling out a victim item's rating from the rest. DV method appends ratings into unrated item cells by the dominant rating in the vector. Such a way of filling unrated item cells does not affect the basic intuition behind this attack. Therefore, it is not expected that increasing δ_{AP} with DV would not make a prominent effect on reconstruction results. On the other hand, RF method randomly fills unrated item cells.

This way of filling unrated cells breaks the alienated status of the victim item. A declining trend is expected for growing δ_{AP} with RF similar to *acting as an active user* attack. Remember that *perfect match* attack dispatches a query and tracks peers who have a perfect match with the dispatched query. Appending new ratings into AP's rating vector does not alter the basic principle of the attack, AP continues to monitor perfect matches in an unaffected way because this protocol only modifies AP's rating and AP knows that there is no data hiding in peers' sites. Appended ratings into a vector could either create a new perfect match or spoil a current perfect match. Therefore, the number of captured perfect matches might stay similar in size. As a result, similar results are expected as δ_{AP} grows with RF and DV.

Table 5.1 displays the results. Note that G is set 1 to eliminate any unexpected consequences that might result from RRT with multi-group when HRI is applied. If the filling method is no filling and δ_{AP} is $0d$ in the table, then it means that this algorithm has been applied as a plain P2P CF (no privacy). In CF mode, *prec* varies between 0.831 and 0.915 and *rec* varies between 0.405 and 0.443 for different attack types. Being able to recover with decent *prec* and *rec* is important if this scheme is utilized in CF setting with no privacy. Remember that AP has no clue about the value of q in peers' vectors and auxiliary information is utilized to speculate the value of q at peers' site. This experimental result with CF is important to stress that exploiting such auxiliary information could be very useful to be exploited with the attacks.

Table 5.1 clearly shows that increasing δ_{AP} values have a dramatic effect in terms of reconstruction for *acting as an active user* attack. One can notice in the table that larger values of δ_{AP} diminish *prec* results for both of the filling methods. In CF settings, *prec* is as high as 0.831, and there is a sharp decrease as soon as privacy measures are introduced with $\delta_{AP} = 0.125d$. As δ_{AP} is increased up to $1d$, the decline continues for DV and RF. Although the decline in *rec* is significant as well, it is not as much affected as *prec*. In the worst case, where δ_{AP} is $1d$, *rec* is recorded around 0.246 and 0.225 for DV and RF, respectively. Results for *acting as an active user* attack are in accordance with the expectation with dramatic declines especially in terms of *prec*. *Alienate the victim* attack displays a constant trend in terms of *prec* and *rec* results for growing δ_{AP} with DV. As stated in this section, when DV is utilized as the filling method, increasing δ_{AP} has no effect on this attack because it has no particular damage in the alienated state of the victim item. Therefore, results for DV with larger δ_{AP} confirms this argument once again.

Table 5.1. *Reconstruction with varying δ_{AP} and filling methods*

Filling Method	δ_{AP}	Alienate the victim		Perfect match		Acting as an active user	
		prec	***Rec***	***prec***	***rec***	***Prec***	***rec***
No filling	0*d*	0.842	0.443	0.915	0.405	0.831	0.443
DV	0.125*d*	0.843	0.445	0.914	0.411	0.107	0.267
	0.25*d*	0.844	0.446	0.913	0.416	0.092	0.260
	0.5*d*	0.842	0.445	0.913	0.422	0.077	0.243
	1*d*	0.844	0.445	0.913	0.436	0.063	0.246
RF	0.125*d*	0.374	0.367	0.913	0.401	0.117	0.278
	0.25*d*	0.250	0.315	0.904	0.396	0.101	0.266
	0.5*d*	0.158	0.245	0.897	0.386	0.085	0.241
	0.1*d*	0.100	0.171	0.891	0.369	0.071	0.225

However, when RF is utilized to fill unrated items' cells, a steep decline is recorded as δ_{AP} gets larger. *Perfect match* attack maintains a stable record in terms of both metrics. Especially, when DV is applied, this attack almost remains defiant for larger δ_{AP} values. Even there is a consistent slight increase in *rec* through δ_{AP} = 1*d*. When RF is utilized, very high *prec* and *rec* values are recorded. However, they are relatively lower than the CF setting. This might be due to the fact that DV is intuitively more inclined to be in harmony with peers' ratings. As a result, increasing δ_{AP} does not hinder AP from discovering perfect matches regardless of the filling method; therefore, evaluation metrics follow a more constant trend for DV and RF.

5.3.2. Effects of varying G

In addition to data hiding by HRI, AP masks the rating vector by utilizing RRT method as well. In data masking, AP generates a uniformly random group number for each peer, which is G_i. G_i prevents a peer from realizing if the related group is indeed preserved or reserved. As G_i grows for a peer, the rating vector is split more. In *acting as an active user* attack, the intuition is that growing G_i values could introduce some improvement on the reconstruction. It is important for this attack to be successful that subsequent queries are only different by one item owing to the fact that temporal changes are monitored between subsequent queries. As G_i gets larger, the size of each group shrinks. As a result, the possibility of appending a rating into a group decreases. Therefore, a group might remain unaltered after HRI, which would help the reconstruction accuracy. It is anticipated that *alienate the victim* attack remains unaffected from larger group sizes if DV is exploited as the filling method. This attack performs well as long as the victim item remained isolated from the rest. Increasing G_i has no effect on the victim item to take away its isolation status. In terms of *perfect match* attack,

introducing larger groups, principally, should have no adverse effect on the basic principle of the attack. Perfect matches can still be captured. However, the number of perfect matches is expected to rise because fewer items need to be matched with corresponding peers' items for a perfect match. The increase in the number of perfect matches could contribute to the reconstruction criteria.

Figure 5.1 displays the *prec* and *rec* results in a two-column graph. In Figure 5.1a and Figure 5.1b, a slightly increasing trend is recorded for *acting as an active user* attack with regards to *prec* and *rec* although *prec* is very low when compared with other attacks. This approves the intuition about larger groups makes it more difficult to append new ratings into each group. Some groups could remain same after HRI. The second attack type, *alienate the victim*, increasing G_i has no effect on the victim item's probability results returned from other peers. Both metrics follow a constant trend for all values of G_i as initially hypothesized. *Prec* metric for perfect match attack shows a moderate increase in between one-group and three-groups. Remaining groups larger than 3 continue a relatively steady trend for *prec*. However, a marked decline is obvious in *rec* metric in the transition from one-group to three-group. This declining trend gradually continues up to the twenty-group scheme. Contrary to the initial thought, downward-trend in *rec* can be attributed to the increase in the number of perfect matches as well. Since CF data sets are usually sparse, perfect matches are usually expected to have a small number of corresponding rated items. These items should be identical or opposite to each other, so they should be marked *dislike* or *like* based on the value of *q*. However, AP does not know

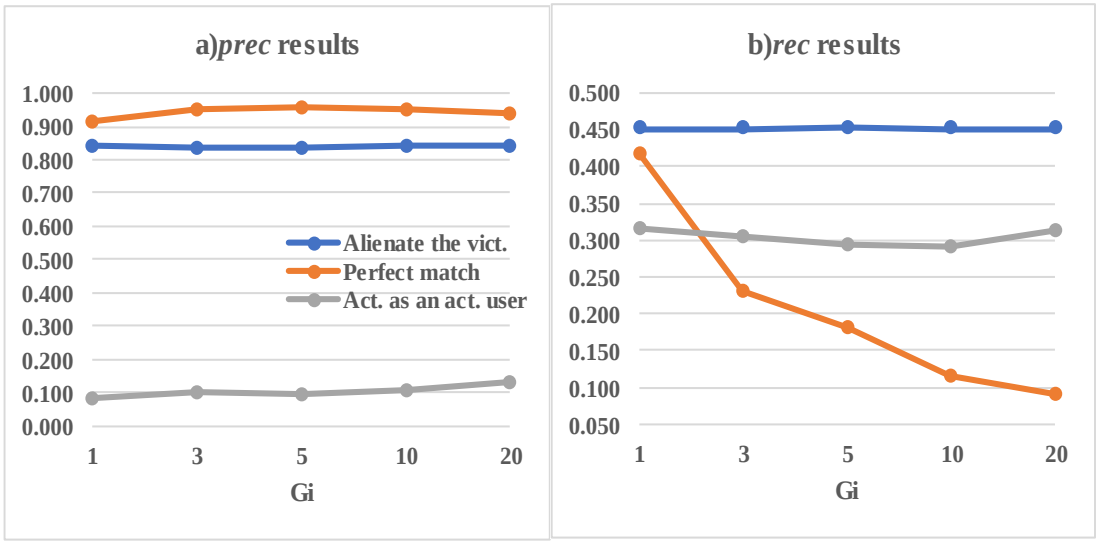


Figure 5.1. *Reconstruction with varying G_i*

which items rated by peers; therefore, she marks all items in the perfect match *?dislike ?like* as discussed in Chapter 4.2. As the number of perfect matches increases, items that do not have any corresponding item in peers' vectors will eventually be marked as unrated if an opposite marking is encountered. Such an incident will cause a decline in *rec* results because more and more items are marked unrated because of the overwhelmingly increased number of perfect matches due to larger groups. *Prec* measures how much of recovered items are indeed identical to the original, so it is not affected by this issue as *rec*.

5.3.3. *Effects of peer privacy*

Up to now, privacy is always viewed from AP's point of view. However, peers can also apply some privacy measures to prevent possible data disclosure. Recall that a peer must rate *q* to join a prediction process. PPP lets peers participate in the prediction process by a random determiner. Therefore, half of the peers join the prediction process without the need to have rated *q*. Non-rater peers determine a rating for *q* based on their default vote. This is a main problem for the malicious AP. Although AP does not know ratings made for *q* by peers even if PPP is not applied, AP makes an assumption that the rating of *q* would be correlated to the average rating collected from IMDB. Since a non-rater peer fills *q* based on her default voting, it adds another uncertainty to all attacks in the reconstruction process. AP marks items' ratings based on *q*'s anticipated value which is the average rating collected from IMDB. Furthermore, PPR lets peers mask their ratings by utilizing HRI. Since appending ratings turns the original data into another one, it is expected that the attacks will be negatively affected in terms of reconstruction results. Figure 5.2 displays the related results. In Figure 5.2, PPP or PPR shows that only PPP or PPR is applied, respectively. PPP&PPR shows that both of PPP and PPR are applied. Reference column displays an experimental setting where the mere difference is that peer privacy measures are not applied.

It is clear in Figure 5.2 that PPP causes a decline in reconstruction metrics for all attack types. The factor causing this decrease could be due to the default vote given for *q* by non-rater peers as discussed. In addition to this factor, in *acting as an active user* attack, AP exploits changes in peers' probability values for subsequent queries. Since participating peers constantly change due to PPP, some peer' probabilities might not be matched with whom AP is monitoring to exploit in the next query. This could be named

as another factor that has an effect on the decline recorded for *acting as active user* attack. The most noticeable point in Figure 5.2a and 5.2b is the decline recorded for *perfect match* attack when PPP is applied. AP marks all items in its rating vector based on *q* in *perfect match* attack; however, the first two attacks deal with only one item, the manipulated or victim item, on each run of the attack. The relatively greater decline in *perfect match* attack could be explained due to the larger item set that this attack dealt with each time.

The main reason that makes sense about the decline in the results for all attack types when PPR is applied is that original data held by peers is masked. In parallel to this notion, the decline is observable for *acting as an active user* and *alienate the victim* attacks. However, *perfect match* attack seems to be resilient to PPR when compared to the reference setting. Similar to the first experiment where δ_{AP} and the filling methods are employed, there are two cases in PPR scenario affecting the number of perfect matches. The first is that a possible perfect match could be lost due to appended ratings. In another case, a new perfect match can be captured. AP and a peer might have no common ratings between each other. Any appended ratings could create a new perfect match. Because some perfect matches are lost and some are gained, this factor is the reason why perfect match attack is resilient to PPR.

When PPP and PPR are applied together, *alienate the victim* and *acting as an active* user attacks records the worst cases while *perfect match* attack records better results than PPP. To summarize, the reference case, where extreme privacy measures are not applied, are always best, but *alienate the victim* attack performs very close rates, and it can be considered reliable compared with two other attacks. *Perfect match* attack perfroms ery

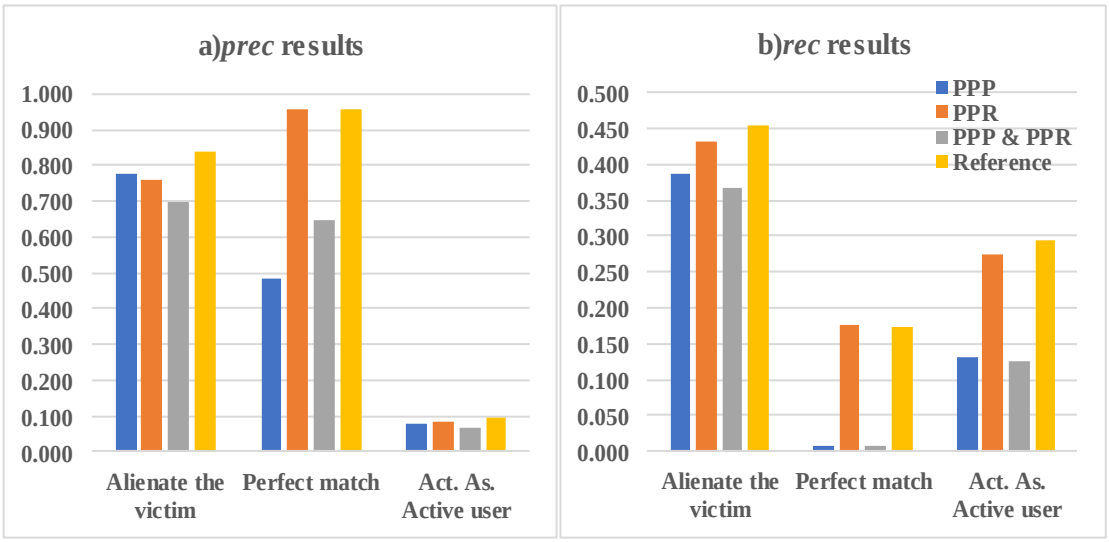


Figure 5.2. *Reconstruction with peer privacy*

poor when PPP is applied, and *acting as an active user* attack records very poor *prec* results when extreme privacy measures are applied.

5.4. Conclusion

In this chapter, three different attack techniques have been examined for an NBC-based binary P2P PPCF scheme. This scheme handles privacy in both AP's and peers' perspective. AP privacy is examined with varying δ_{AP} and the filling methods while peers' privacy is examined in terms of PPP and PPR. Experimental results show that *acting as an active user* attack is not successful for increasing δ_{AP} values in terms of particularly precision metric. While *alienate the victim* attack is resilient to large δ_{AP} if DV is utilized as the filling method. However, this attack records a dramatically declining trend when RF is applied as the filling method. *Perfect match* attack presents almost a stable trend for increasing δ_{AP} for both of the filling methods. When the number of groups is increased, *acting as an active user* attack performs some improvements; however, its precision results are already too low to be considered applicable. On the other hand, *alienate the victim* attack performs a stable trend in terms of precision and recall. *Perfect match* attack is very resilient to protect its precision value for larger groups; nevertheless, it demonstrates a steep decline in terms of recall. The decline in the recall might be because of the increasingly large number of perfect matches. When peers apply privacy metrics, which is primarily designed to prevent from *acting as an active user* attack, *alienate the victim* attacks seems to be very promising in terms of PPP and PPR. In terms of PPR, *perfect match* attack displays very consistent results compared to the reference setting where no peer privacy is utilized.

To sum up, privacy measures taken by targeted PPCF scheme (Kaleli and Polat, 2010) is very successful to thwarting *acting as an active user* attack. However, *alienate the victim attack* can reconstruct with very high precision unless RF is utilized. Results show that RF is a must to prevent peers from this attack. *Perfect match* attack also reconstructs with very high precision unless peers decide to protect themselves by PPP. However, utilizing PPP could harm prediction results as well because it lets peers take part in the prediction without rating *q*. Since *alienate the victim* and *perfect match* attacks are very resilient under different privacy settings, extra measures apart from RF and PPP should be developed. This study also confirms that exploiting auxiliary information would be very crucial to infer confidential data. NBC-based P2P binary PPCF scheme

(Kaleli and Polat, 2010) should be investigated in detail to take it one step further in terms of attack types given in this study as future work.

6. CONCLUSION

In this dissertation, various attack techniques are discussed to derive original binary ratings from perturbed data when privacy preserving collaborating filtering schemes with different data partitioning scenarios are applied. These attack techniques are experimentally tested with a well-known benchmark data set in terms of two different yet related evaluation metrics, precision and recall, to analyze how much of the original binary data can be recovered when privacy measures and other parameters are varied. Based on the experiments performed throughout the dissertation, conclusions can be listed as follows.

The central server-based privacy preserving collaborating filtering scheme utilizes randomized response technique with the multi-group approach. It is shown that it can be reconstructed with decent accuracy. Since randomized response technique is designed to allow an interviewer to estimate the percentage of a sensitive attribute in a population, a malicious data holder can easily estimate the percentage of *like*s and *dislike*s for an item. The data holder can exploit estimated percentages of each item to create a list of extreme items, which are voted either *like* or *dislike* by the majority of users. The basic idea behind deriving private ratings of users is built upon checking these extreme items in perturbed user vectors to validate whether the related user rates the extreme items in accordance with their possible ratings estimated from the perturbed data. If the majority of extreme items are rated identically with their possible ratings, then the master party could deduce that the related user persevered her rating vector. Otherwise, the rating vector is reversed. Four important points can be drawn from experimental results. First, experimental results show that a malicious data holder could reconstruct with very high precision results with increasing number of extreme items up to a certain point when θ is predetermined and at a moderate level such as 0.650 not to sacrifice prediction accuracy. Second, when θ is varied between 0.510 and 0.950, where the randomness reaches its peak at 0.510, the precision results decline, but it is always better than the expected precision of $\theta \times 100$. Likewise, when the number of groups, which is the other privacy measure, is varied, the reconstruction results are in decline; however, it is again always greater than the expected precision. Based on findings from the second and the third, a malicious data holder is sure that more private original ratings can be derived no matter how tight the privacy measures are set. Last, auxiliary information is also integrated to improve the reconstruction results. The experimental results show that integrating auxiliary information improves the

reconstruction when the number of extreme items are small. However, the improvement achieved by auxiliary information is marginal when the number of extreme items is sufficiently large for reconstruction. Hence, a malicious data holder could utilize a small set of extreme items with auxiliary information to reach similar reconstruction results.

In addition to data masking by randomized response method with multi-group, the central server-based privacy preserving collaborating filtering scheme hides whether an item is rated or not by inserting fake ratings into unrated items' cells up to the percentage of the density of the related user vector. Since fake ratings are appended randomly, the use of auxiliary information is preferred to identify genuine ratings. The idea behind utilizing auxiliary information to derive genuine items is to exploit reliable data sources can reveal much about the targeted data set. Therefore, auxiliary information such as the number of votes and movie awards are collected from a well-known, recognized and reliable data source. The related experiment records decent results in terms of precision and recall, respectively. Moreover, precision beats the estimated expected value of 0.666 for data hiding.

When two or multi data holders want to collaborate to enhance their rating matrix while producing recommendations, the primary concern is to preserve the privacy of institutional data. Since data is shared between a different number of parties in privacy-preserving partitioned and distributed collaborating filtering schemes, parties need to exchange partial similarity values to calculate the ultimate similarity value. Therefore, such exchanges between parties are exploited by a malicious party in this study. Three of the four attacks in this dissertation exploit such similarity values exchanged between parties to reconstruct private institutional data. Although reconstruction results vary with different control parameters in the experiments, such attacks pose a high risk for parties. For example, *alienate the victim* attack can achieve full precision and recall results if data is vertical and default voting is utilized for appending fake ratings into unrated items' cells. *Perfect match* attack generally follows a stable trend for varying privacy measures. *Acting as an active user* attack is in a declining trend and not successful in terms of precision with tightened privacy measures but recall values are very high. *knn* attack exploits the recommendation output. On the other hand, experimental results of *knn* attack are too low.

Attacks exploiting similarity values exchanged between parties cannot be applied on Naïve Bayes Classifier based privacy preserving collaborative filtering scheme with

horizontal distributed data. The master party has to know the value of the queried item for reconstruction with these attacks. However, the queried item resides at collaborating parties if data is horizontally distributed between parties. A modification is necessary, and this bottleneck is overcome by utilizing auxiliary information. By using auxiliary information, the value of the queried item is estimated to activate the attacks. The last data partitioning scheme in this dissertation is a peer to peer scheme with Naïve Bayes Classifier for predictions. With peer to peer collaboration, peers eliminate the reign of data holders while producing collaborative filtering with privacy. Since partial conditional probability values to calculate the final similarity are exchanged between peers, this scheme is also prone to all attacks that can be applied on distributed schemes. However; the value of the queried item is not known by the malicious active peer because peer to peer collaboration is inherently a horizontal scheme in terms of data partitioning. Similar to multi-party horizontally distributed schemes, auxiliary information is utilized to estimate the value of the queried item. The use of auxiliary information is shown to be very important while disclosing private information throughout the dissertation. It is first exploited to derive actual rating values and rated items with central server-based schemes; then it is exploited to derive institutional and individual private information from horizontally distributed data schemes and peer-to-peer privacy preserving collaborating filtering schemes. This study only integrated item-related auxiliary information; nonetheless, auxiliary information could also be linked to users. The popularity of social media is undeniable, and it is a great opportunity to link users to know more about them. Such an integration of auxiliary information could be more destructive while deriving confidential information from privacy enhanced systems.

To sum up, the main concentration in this dissertation is to derive private binary ratings from different privacy preserving collaborating filtering schemes when data is centrally stored by a data holder, distributed between parties or peers. These privacy-preserving collaborative filtering schemes are scrutinized in terms of privacy measures, and various attacks are executed to exploit any deficiency to recover confidential user, institutional or peer data. Additionally, auxiliary information is utilized to show that some privacy measures or structural bottlenecks can be circumvented. The problem with the central server-based binary privacy preserving collaborating filtering scheme is that the randomized response technique, which is used by users to perturb their data, discloses itself. With distributed and peer to peer schemes are investigated, the main deficiency is

the similarity values exchanged between parties or peers, respectively. Additionally, splitting a vector into groups seems to be effective in practice, but experiments disprove this.

When the computational power of modern day devices is taken into account, peer to peer collaboration should be more focused to offer private recommendations as future work. Especially, a solution is needed to communicate partial similarity values. Partial similarity values should not be transferred to the active peer without any modification, or some mediator peers could be selected randomly to transfer aggregate partial similarities to confuse the active peer. Therefore, the future work will focus more on the peer-to-peer collaboration to produce private recommendations.

REFERENCES

Ackerman, M.S., Cranor, L.F. and Reagle, J. (1999). Privacy in e-commerce: examining user scenarios and privacy preferences. *Proceedings of the 1st ACM conference on Electronic commerce*, New York, NY, USA: ACM, pp. 1-8.

Agrawal, D. and Aggarwal, C.C. (2001). On the design and quantification of privacy preserving data mining algorithms. *Proceedings of the 20th ACM SIGMOD-SIGACTSIGART Symposium on Principles of Database Systems*, New York, NY, USA: ACM, pp. 247-255.

Agrawal, R. and Srikant, R. (2000). Privacy-preserving data mining. *Proceedings of the 19th ACM SIGMOD International Conference on Management of Data*, New York, NY, USA: ACM, pp. 439-450.

Bélanger F. and Crossler, R.E. (2011). Privacy in the digital age: a review of information privacy research in information systems. *MIS Quarterly*, 35(4), 1017-1041.

Berendt, B., Günther, O. and Spiekermann S. (2005). Privacy in e-commerce: stated preferences vs. actual behavior. *Communications of the ACM*, 48(4), 101-106.

Berkovsky, S., Eytani, Y., Kuflik, T. and Ricci, F. (2005). Privacy-enhanced collaborative filtering. *In Workshop on Privacy-Enhanced Personalization*, http://isr.uci.edu/pep05/papers/PEPfinal.pdf. (Accessed: 16.10.2017)

Berkovsky, S., Eytani, Y., Kuflik, T. and Ricci, F. (2007). Enhancing privacy and preserving accuracy of a distributed collaborating filtering. *Proceedings of the 2007 ACM conference on Recommender systems*, New York, NY, USA: ACM, pp. 9-16.

Berkovsky, S., Kuflik, T. and Ricci, F. (2012) The impact of data obfuscation on the accuracy of collaborative filtering. *Expert Syst Appl*, 39(5), 5033-5042.

Bilge, A., Kaleli, C., Yakut, I., Gunes, I. and Polat, H. (2013). A survey of privacy-preserving collaborative filtering schemes. *Int J Softw Eng Know*, 23(8), 1085-1108.

Bilge, A. and Polat, H. (2010). Improving privacy-preserving NBC-based recommendations by preprocessing. *Proceedings of 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, Toronto, ON, Canada: IEEE, pp. 143-147.

Breese, J. S., Heckerman, D. and Kadie, C. (1998). Empirical Analysis of Predictive Algorithms for Collaborative Filtering. *Proceedings of the Fourteenth Conference*

on Uncertainty in Artificial Intelligence, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., pp. 43-52.

Calandrino, J.A., Kilzer, A., Narayanan, A., Felten, E.W. and Shmatikov, V. (2011). "You might also like:" privacy risks of collaborative filtering. *Proceedings of 2011 IEEE Symposium on Security and Privacy*, Washington, DC, USA: IEEE, pp. 231-246.

Canny, J. (2002). Collaborative filtering with privacy. *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA : ACM, pp. 238-245.

Chen, K. and Liu, L. (2005). Privacy preserving data classification with rotation perturbation. *Proceedings of the 5th IEEE International Conference on Data Mining*, Washington, DC, USA: IEEE, pp. 589-592.

Chen, K., Sun, G. and Liu, L. (2007). Towards attack-resilient geometric data perturbation. *Proceedings of the 2007 SIAM International Conference on Data Mining*, Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, pp. 78-89.

Clarke, R. (1999). Internet privacy concerns confirm the case for intervention. *Communications of ACM*, 42 (2), 60-67.

Cooley, T. M. (1879). *A treatise on the law of torts or the wrongs which arise independent of contract*. Chicago: Callaghan.

Cranor, L. F. (2003). 'I didn't buy it for myself' privacy and ecommerce personalization. *Proceedings of the ACM Workshop on Privacy in the Electronic Society*, New York, NY, USA: ACM, pp. 111-117.

Culnan, M. J. (1993). "How did they get my name?": an exploratory investigation of consumer attitudes toward secondary information use. *MIS Quarterly*, 17 (3), 341-363.

Demirelli Okkalioglu, B., Koc, M. and Polat, H. (2016) Reconstructing rated items from perturbed data. *Neurocomputing*, 207, 374-386.

Dutta, H., Kargupta, H., Datta, S. and Sivakumar, K. (2003). Analysis of privacy preserving random perturbation techniques: Further explorations. *Proceedings of the 2003 ACM Workshop on Privacy in the Electronic Society*, New York, NY, USA: ACM, pp. 31-38.

Friedman, A., Knijnenburg, P.B., Vanhecke, K., Martens, L. and Berkovsky, S. (2015). Privacy aspects of recommender systems. F. Ricci, L. Rokach and B. Shpira (Eds.), In *Recommender Systems Handbook* (pp. 649, 688). USA: Springer.

Gamb, S. and Lolive, S. (2013). SlopPy: slope one with privacy. *Lect Notes Comput Sc*, 7731, 104-117.

Goldberg, D., Nichols, D., Oki, B. M. and Terry, D. (1992). Using Collaborative Filtering to Weave an Information Tapestry. *Communications of the ACM*, 35 (12), 61–70.

Guo, S. and Wu, X. (2006). On the use of spectral filtering for privacy preserving data mining. *Proceedings of the 2006 ACM Symposium on Applied Computing*, New York, NY, USA: ACM, pp. 622-626.

Guo, S. and Wu, X. (2007). Deriving private information from arbitrarily projected data. *Lect Notes Comput Sc*, 4426, 84-95.

Guo, S., Wu, X., Li, Y. (2006) On the lower bound of reconstruction error for spectral filtering based privacy preserving data mining. *Lect Notes Comput Sc*, 4213, 520-527.

Guo, S., Wu, X. and Li, Y. (2008). Determining error bounds for spectral filtering based reconstruction methods in privacy preserving data mining. *Knowl Inf Syst*, 17(2), 217-240.

Han, J., Kamber, M. and Pei, J. (2012). *Data mining concepts and techniques*. Waltham, MA, USA: Morgan Kaufmann Publishers.

Herlocker, J. and Konstan, J. (1999). An algorithmic framework for performing collaborative filtering. *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA: ACM, pp. 230-237.

Hu, Y., Koren, Y. and Volinsky, C. (2008). Collaborative Filtering for Implicit Feedback Datasets. *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, Washington, DC, USA: IEEE, pp. 263–272.

Huang, Z. and Du, W. (2008). OptRR: Optimizing randomized response schemes for privacy-preserving data mining. *Proceedings of the 24th International Conference on Data Engineering*, Washington, DC, USA: IEEE, pp. 705-714.

Huang, Z., Du, W. and Chen, B. (2005) Deriving private information from randomized data. *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*, New York, NY, USA: ACM, pp. 37-48.

Hyvärinen, A., Karhunen, J. and Oja, E. (2001). *Independent component analysis*. John Wiley & Sons.

International Telecommunication Union (2017). *ICT Facts and Figures 2017*. Geneva, Switzerland: International Telecommunication Union.

Kaleli, C. and Polat, H. (2007a) Providing naïve bayesian classifier-based private recommendation on partitioned data. *Lect Notes Comput Sc*, 4702, 515-522.

Kaleli, C. and Polat, H. (2007b). Providing private recommendations using naïve bayesian classifier. *Adv Soft Comp*, 43, 168-173.

Kaleli, C. and Polat, H. (2009). Similar or dissimilar users? Or both?. *Proceedings of 2009 Second International Symposium on Electronic Commerce and Security*, Nanchang City, China, China, May 22 - 24, 2009, pp. 184-189. ISBN: 978-0-7695-3643-9.

Kaleli, C. and Polat, H. (2010) P2P collaborative filtering with privacy. *Turk J Electr Eng Co*, 18 (1), 101-116.

Kaleli, C. and Polat, H. (2015). Privacy-preserving naïve bayesian classifier-based recommendations on distributed data. *Comput Intell*, 31 (1), 47-68.

Kargupta H, Datta S, Wang Q, and Sivakumar K (2003). On the privacy preserving properties of random data perturbation techniques. *Proceedings of Third IEEE International Conference on Data Mining*, Washington, DC, USA: IEEE, pp 99-106.08

Kargupta, H., Datta, S., Wang, Q. and Sivakumar, K. (2005). Random-data perturbation techniques and privacy preserving data mining. *Knowl Inf Syst*, 7(4), 387-414.

Konstan, J. A., Miller, B. N., Maltz, D., Herlocker, J. L., Gordon, L. R. and Riedl, J. (1997). GroupLens: Applying Collaborative Filtering to Usenet News. *Communication of the ACM*, 40 (3), 77–87.

Liu, K. (2007). *Multiplicative data perturbation for privacy preserving data mining*. Ph.D. thesis, College Park, MD, USA: University of Maryland.

Liu, K., Kargupta, H. and Ryan, J. (2006). Random projection based multiplicative data perturbation for privacy preserving distributed data mining. *IEEE T Knowl Data En*, 18(1), 92-106.

Liu, K., Giannella, C. and Kargupta, H. (2006). An attacker's view of distance preserving maps for privacy preserving data mining. *Lect Notes Comput Sc*, 4213, 297- 308.

Magi, T. (2011). Fourteen reasons privacy matters: a multidisciplinary review of scholarly literature. *The Library Quarterly: Information, Community, Policy*, 81(2), 187-209.

McAfee, A., Brynjolfsson, E., Davenport, T. H., Patil, D. J. and Barton, D. (2012). Big data. The Management Revolution. *Harvard Bus Rev*, 90 (10), 61–67.

Miyahara, K., and Pazzani, M. J. (2000). Collaborative filtering with the simple Bayesian classifier. *Lect Notes Comput Sci*, 1886, 679-689.

Oard, D.W. and Kim, J. (1998). Implicit feedback for recommender systems. *Proceedings of the AAAI workshop on recommender systems*, Menlo Park, CA, USA: AAAI Press, pp. 81-83.

Oliveira, S.R.M. and Zaïane, O.R. (2010) Privacy preserving clustering by data transformation. *Journal of Information and Data Management* 1 (1), 37-52.

Ozturk, A. and Polat, H. (2010). From existing trends to future trends in privacy-preserving collaborative filtering. *WIREs Data Min Knowl*, 5(6), 276-291.

Paine, C., Reips, U.D., Stieger, S., Joinson, A., and Buchanan, T. (2007). Internet users' perceptions of 'privacy concerns' and 'privacy actions'. *Int J Hum-Comput Stud*, 65(6), 526-536.

Polat, H. and Du, W. (2003). Privacy-preserving collaborative filtering using randomized perturbation techniques. *Proceeding of Third IEEE International Conference on Data Mining*, Melbourne, FL, USA, Nov. 19-22, 2003, ICDM '03. pp. 625-628. ISBN: 0-7695-1978-4. doi: 10.1109/ICDM.2003.1250993

Polat, H., and Du, W. (2005a). Privacy-preserving collaborative filtering. *Int J Electron Comm*, 9 (4), 9-35.

Polat, H., and Du, W. (2005b). Privacy-preserving collaborative filtering on vertically partitioned data. *Lect Notes Comput Sc*, 3721, 651-658.

Polat, H., and Du, W. (2005c). Privacy-preserving top-N recommendation on horizontally partitioned data. *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence*, Washington, DC, USA: IEEE, pp. 725-731.

Polat, H. and Du, W. (2005d). SVD-based collaborative filtering with privacy. *Proceedings of the 2005 ACM symposium on Applied computing*, New York, NY, USA: ACM, pp.791-795.

Polat, H. and Du, W. (2006). Achieving private recommendations using randomized response techniques. *Lect Notes Comput Sci*, 3918, 637–646.

Polat, H. and Du, W. (2007). Effects of inconsistently masked data using RPT on CF with privacy. *Proceedings of the 2007 ACM symposium on Applied computing*, Seoul, Korea, March 11 - 15, 2007, SAC '07. pp. 649-653. ISBN: 1-59593-480-4.

Polat, H. and Du, W. (2008). Privacy-preserving top-N recommendation on distributed data. *J Assoc Inf Sci Tech*, 59 (7), 1093-1108.

Polatidis, N., Georgiaidis, C.K., Pimenidis, E. and Mouratidis, H. (2017). Privacy-preserving collaborative recommendations based on random perturbations. *Expert Syst Appl*, 71, 18-25.

Resnick, P. and Varian, H. (1997). Recommender Systems. *Communications of ACM*, 40 (3), 56-58.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. and Riedl, J. (1994). GroupLens: An Open Architecture for Collaborative Filtering of Netnews. *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, New York, NY, USA: ACM, pp.175-186.

Sang, Y., Shen, H. and Tian, H. (2009). Reconstructing data perturbed by random projections when the mixing matrix is known. *Lect Notes Comput Sc*, 5782, 334-349.

Sang, Y., Shen, H. and Tian, H. (2012). Effective reconstruction of data perturbed by random projections. *IEEE T Comput*, 61(1), 101-117

Sarwar, B.M., Karypis, G., Konstan, J.A., and Riedl, J. (2001). Item-based Collaborative Filtering Recommendation Algorithms. *Proceedings of the 10th International Conference on World Wide Web*, Hong Kong, May 1 - 5, 2001, WWW '01. pp. 285–295. ISBN:1-58113-348-0.

Sarwar, B. M., Konstan, J.A., Borchers, A., Herlocker, J., Miller, B. and Riedl, J. (1998). Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system. *Proceedings of the 1998 ACM Conference on Computer Supported Cooperative work*, New York, NY, USA: ACM, pp. 345-354.

Schafer, J. B., Frankowski, D., Herlocker, J. and Sen, S. (2007). Collaborative Filtering Recommender Systems. *Lect Notes Comput Sc*, 4321, 291–324.

Solove, D. J. (2002). Conceptualizing privacy. *Calif Law Rev*, 90(4), 1087-1155.

Su, X. and Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Lect Notes Artif Int*, 2009, 1-9.

Turgay, E.O., Pedersen, T.B., Saygin, Y., Savas, E. and Levi, A. (2008) Disclosure risks of distance preserving data transformations. *Lect Notes Comput Sc*, 5069, 79-94.

Warner, S.L. (1965). Randomized Response: A Survey Technique for Eliminating Evasive Answer Bias. *J Am Stat Assoc*, 60 (309), 63-69.

Warren, S. D. and Brandeis, L. D. (1890). The Right to Privacy. *Harvard Law Rev*, 4(5), 193–220.

Westin, A. F. (1967). *Privacy and Freedom*. New York: Atheneum as cited in Solove, D. J. (2002). Conceptualizing privacy. *Cal Law Rev*, 90(4), 1087-1155.

Yakut I. and Polat, H. (2010). Privacy-preserving SVD-based collaborative filtering on partitioned data. *Int J Inf Tech Decis*, 9 (03), 473-502.

Zhang, S., Ford, J., and Makedon, F. (2006). Deriving private information from randomly perturbed ratings. *Proceedings of the 2006 SIAM International Conference on Data Mining*, Bethesda, MD, USA, Apr. 20-22, 2006. pp.59-69. ISBN: 978-0-89871-611-5. J. Ghosh, D. Lambert, D. Skillicorn and J. Srivastava (Eds.).